The $\mathbf{F_L eX^p_a R_T}$ Particle Dispersion Model Version 6.1

User Guide

ANDREAS STOHL

Norwegian Institute for Air Research,
Department for Regional and Global Pollution Issues
P.O. Box 100, 2027 Kjeller, Norway

PETRA SEIBERT

Institut für Meteorologie und Physik,
University of Agricultural Sciences Vienna
Türkenschanzstrasse 18, 1180 Vienna, Austria

CAROLINE FORSTER

Norwegian Institute for Air Research,
Department for Regional and Global Pollution Issues
P.O. Box 100, 2027 Kjeller, Norway

March 15, 2005

Contents

Chapter 1

Introduction

Particle modeling is the Lagrangian approach towards the simulation of atmospheric dispersion. Lagrangian particle models compute trajectories of a large number of individual so-called particles (not necessarily representing real particles, but infinitesimally small air parcels) to describe the transport and diffusion of substances in the atmosphere. Usually they are used to simulate the dispersion of air pollutants, released for instance due to an accident in a nuclear power plant, but they are suitable also for a very wide range of other applications.

A main advantage of Lagrangian models is that there is no artificial numerical diffusion such as it occurs in Eulerian models. This is of special importance in the vicinity of the source of an air pollutant, where in Eulerian models the pollutant is instantaneously mixed over at least one grid box, which can cause large subsequent transport errors. The description of turbulence with particle models is of similar quality as large eddy simulations, and it is clearly more accurate than with Eulerian models. Particle models are independent of a computational grid and, in principle, provide infinitesimally small resolution. Practically, however, their effective resolution is determined by the number of particles released and indirectly also by the resolution of meteorological input data.

Tens or hundreds of thousand particles may be required to accurately describe a multi-day transport episode from a point source, and even more so for large area sources. This posts high demands on computer resources and explains why particle models have only become popular in recent years since powerful computers became available. The basis for most of current atmospheric particle models was developed in a paper by Thomson (1987). A very thorough monograph on the theoretical background of stochastic Lagrangian models has been published by Rodean (1996). Another interesting paper is the one by Wilson and Sawford (1996). Reviews of particle modeling, which also describe practical aspects,

were provided by Zannetti (1992) and Uliasz (1994).

FLEXPART is a Lagrangian particle dispersion model designed for both operational emergency response and research applications. It simulates the long-range transport, diffusion, dry and wet deposition, and radioactive decay of air pollutants released from point, line, area or volume sources, but can also be used in domain-filling modes where the entire atmosphere is subdivided in particles, each representing an equal part of the atmospheric mass. FLEXPART can be used in forward mode to simulate the dispersion of pollutants released at a location, or in backward mode to determine potential source contributions for a given receptor. Forward trajectories are calculated in the forward mode, backward trajectories in the backward mode. The management of input data was largely taken from FLEXTRA, a kinematic trajectory model (Stohl *et al.,* 1995) developed at the Institute of Meteorology and Physics of the University of Agricultural Sciences in Vienna. The basic transport and diffusion code of the model was developed during the first author's military service at the NBC school of the Austrian Forces (version 1.x). The deposition code was developed later on as a contract work for the Central Institute for Meteorology and Geodynamics in Vienna (version 2.x). Work on FLEXPART was then continued at the University of Munich and at the Technical University of Munich. The code underwent a major revision to version 3.0, which optimized the runtime performance, especially when using short time steps. Some minor bugs were found and removed, the numerics of the model has been improved, and a density correction was introduced. Further updates included a convection scheme (version 4.0), and better backward calculation capabilities as well as improvements in the input/output handling, etc. (version 5.0). While many changes were made in the years since version 5.0 was released, both during the first authors' stay at the NOAA Aeronomy Laboratory and now, by Caroline Forster and the first author at NILU, it took some time to release again a fully documented version, now called version 6.0.

The reference version of FLEXPART that is described here is based on model level data of the numerical weather prediction model of the European Centre for Medium-Range Weather Forecasts (ECMWF). Gridded data of this model are currently available at resolutions up to better than 0.5 ° longitude and latitude, 60 vertical levels, and with a frequency of 3 hours. There are other data sources to drive FLEXPART , i.e., output from the National Center of Environmental Prediction's (NCEP) Global Forecast System (GFS) model or from the MM5 model, but the FLEXPART versions employing these data are modified and deviate in some aspects from the one described here. Other users have also developed interfaces to other data sources.

FLEXPART versions 1.x and 2.x were evaluated using data from practically all large-

scale tracer experiments available, namely from the Cross-Appalachian Tracer Experiment (CAPTEX), the Across North America Tracer Experiment (ANATEX) and the European Tracer Experiment (ETEX), comprising a total of 40 usable releases (Stohl *et al.,* 1998). FLEXPART performed rather well compared to other models. Version 3.1 was again validated with the CAPTEX and ETEX data. Version 3.2, which changed little in the model physics, only with the ETEX data. The performance was similar to previous versions.

More recent comparisons of versions 3.x and 4.x with measurement data were made in the context of studies of intercontinental pollution transport from North America to Europe (Stohl and Trickl, 1999; Forster et al., 2001; Spichtinger et al., 2001; Stohl et al., 2002a). The last study also validated FLEXPART backward simulations. A special FLEXPART version describing the transport of a stratospheric ozone tracer was validated with measurement data from mountain stations (Stohl et al., 2000). These features are now also available in version 6.0. Currently, Caroline Forster is validating version 6.0 again with CAPTEX data, and studies of pollutant transport are ongoing that provide a validation of the model.

Chapter 2

New features

2.1 New features of version 3.1

The model physics has changed little since version 3.0. However, FLEXPART 3.1 has a few attractive new features, improving it's usability:

1. Global datasets allowed: Use of cyclic boundary conditions; polar stereographic projection near the poles.

2. Nesting: An unlimited number of nests at various nesting levels may be used.

3. Uncertainty output: For every grid cell, FLEXPART calculates the uncertainty caused by the limited number of particles being used for the simulation. Summary uncertainty statistics are continuously created during the model run, so that one can re-start the model with more particles at an early stage, if necessary.

4. Sparse matrix format output: For every output field, it is checked whether it is more efficient to save the field in sparse matrix format, which only contains the concentrations in grid boxes with non-zero concentrations, rather than to dump the whole field. This saves a lot of mass storage space, especially for rather short model runs (small tracer clouds).

5. Alternative output of mixing ratio: one can choose whether the output shall be in concentration (ng m^{-3}, Bq m^{-3}) or in mixing ratio (pptv) units or in both.

6. Additional optional direct output of particle positions, which may be useful for visualization. Also, particles may be dumped at the end of the simulation, and may be used for re-initialization of a subsequent model run.

7. Tested for inverse modeling: FLEXPART 3.1 now definitely also works backward in time, which is useful for inverse modeling purposes. A first attempt at the reconstruction of the ETEX source term undertaken by Seibert and Stohl [2000] shows promising results.

8. Subgrid variability of mixing heights is treated somewhat differently than before.

2.2 New features of version 3.2

1. FLEXPART can now create age spectra of tracers. The age of a particle is defined by the time passed since its release. From the age information carried by all the particles, not only total concentrations are calculated, but optionally they can be split into the contributions of several age classes as defined by the user.

2. Calculation of mass fluxes: optionally, the east-west, north-south and up-down gross mass fluxes across the centerlines of each cell of the output grid can be calculated, also for age classes.

3. For backward model runs, the "concentration output" is not in mass/volume, but in residence time $\times$ mass of the release, as needed by inversion procedures.

4. Particle position output from a previous run can be used to re-initialize particles in the present run. This is a useful option for longer model runs (e.g. when not all wind fields can be kept on disc). A re-start can also be used to remove "expired" particles from the memory.

5. Easy switching between "Little Endian" and "Big Endian" computers just by changing one parameter in file `includepar`.

6. Input can, alternatively, be in free format. Suitable for the experienced user.

2.3 New features of version 4.0

1. The preliminary convection scheme of version 3.2 has been replaced by the convection scheme of Emanuel and Živković-Rothman (1999).

2.4 New features of version 5.0

1. The output has been re-arranged. Now, individual files are generated for every date. This allows much faster access to specific output dates, e.g., for plotting the results.

2. The heights of release points can be specified in file `RELEASES` either in meters above ground level or in meters above sea level. Note that the format of the file `RELEASES`, thus, has changed.

3. Improved backward calculation capabilities: the output now has the unit of seconds and is corrected for differences in density between source and receptor (see Seibert, 2002); the output fields in backward runs are dimensioned with maxpoint instead of maxspec (which must be 1 for backward runs). This allows simultaneous backward calculations for a much larger number (order of hundreds) of receptors than was possible to date.

4. Improved warm start option: If "particle dump at end" is selected, particle output is generated (but overwritten) at every output time. That way, a warm re-start is always possible from the last available output date (useful if a long model run is interrupted).

5. Plume trajectories can be calculated using cluster analysis as described in Stohl et al. (2002b). This method preserves a maximum of information from the $\mathrm{FLEXPART}$ simulation in a minimum of output. It is especially useful for simulations of many releases, for which a full dump of the output would exceed hardware limitations or would be too large for subsequent statistical analyses.

6. Potential vorticity (3-d field) and tropopause height (2-d field) are calculated from winds and temperature. They are currently only used in conjunction with the plume trajectories, in order to determine how many particles reside in the stratosphere.

7. The uniform kernel is not used during the first 3 hours after a particle's release (when it is attributed only to the grid cell it resides in), in order to avoid artificial smoothing of concentration fields close to the source.

8. Global mother grids can be shifted such that the left grid boundary is at any desired longitude. That way, nested wind fields or output grids overlapping the original domain "boundary" can be accomodated.

9. The efficiency of memory usage has been improved. The storage space occupied by "expired" particles (because their maximum age has been reached) is made available for new particles. In case of continuous sources less memory is needed if the simulation period is longer than the maximum age considered. This way, continuous simulations are possible even over periods of a year or more.

10. In order to improve numerical accuracy, the trajectory integration is done now with one iteration of the Petterssen scheme for the grid-scale winds whenever possible.

11. The scaling factor used for mesoscale velocity fluctuations has been reduced to 0.16 (from 0.5) of the standard deviation of the winds at the surrounding grid points.

12. An outdated option to provide the topography and land/sea mask separately from the other ECMWF data has been removed, in order to shorten the code.

13. The number of model levels is not doubled anymore in the process of transformation to terrain-following coordinates.

2.5 New features of version 6.0

1. The convection scheme has been revised completely. While the underlying scheme is still the one of Emanuel and Živković-Rothman (1999), particle redistribution is now completely stochastic. This eliminated the need for the Eulerian tracer displacement and reduced computation time dramatically, especially when many particles are used. Convection is now also called every $\mathrm{F_L eX^p_a R_T}$ model time step.

2. Turbulence above the boundary layer is now based on a diffusivity scheme that distinguishes between troposphere and stratosphere.

3. The input coordinate for particle releases can now be meters above sea level, meters above ground level, or pressure.

4. $\mathrm{F_L eX^p_a R_T}$ can now also output one nested field with higher resolution.

5. Particle releases (i.e., emission fluxes) can now have a specified daily and/or weekly cycle.

6. A new feature is that $\mathrm{F_L eX^p_a R_T}$ can be used in a domain-filling option. In this option, the domain is "filled" with particles of equal mass, distributed according to

the density distribution in the atmosphere. These particles are then advected in the atmosphere. If a limited domain is used, particles are "created" at the inflowing boundary and "destroyed" at the outflowing boundary.

7. The above feature can also be used to simulate the transport of a stratospheric ozone tracer with ozone values taken at the boundaries from an ozone/potential vorticity relationship.

8. For backward runs, the input can either be a concentration or a mixing ratio.

9. Particle positions can now also be output in a very compact format. If this option is chosen, all particles receive an individual number (instead of being marked by release point) in order to identify continuous trajectories in the particle output.

2.6 New features of version 6.1

1. The horizontal advection part of FLEXPART was changed from single to double precision. Differences to single precision were found to be very small in the examples studied. But it is expected to increase the numerical accuracy, especially when short time steps are used, i.e., when the addition of position change (a small number) to position (a relatively large number) becomes inaccurate.

2. A number of small coding changes, especially in the interpolation routines, led to an about 10% faster code (without convection).

3. Due to often relatively large values of the unsaturated downdrafts in the convection scheme, the compensating subsidence was found to be occasionally directed upwards. This seems to be unrealistic and also caused numerical problems. Therefore, unsaturated downdrafts were taken out of the redistribution matrix and combined with the subsidence velocity. FLEXPART now produces a numerically highly accurate redistribution of particles due to convection. With a large enough number of particles, the vertical distribution of mass within the atmosphere is now almost exactly conserved.

Chapter 3

Input data and grid definitions

FLEXPART is based on gridded meteorological fields (analyses or forecasts) from the numerical weather prediction model of the ECMWF (ECMWF, 1995) given in a latitude/longitude coordinate system. The data are assumed to be in GRIB format. The GRIB decoding software is *not* provided with FLEXPART . GRIB decoding libraries, thus, must be available on the computer where FLEXPART shall be installed. The GRIB software is available from ECMWF or NCEP (see links from the FLEXPART webpage). Note that there are differences between the different existing GRIB versions, concerning both parameter codes and input field organisation. FLEXPART follows the rules used at ECMWF.

3.1 The files includepar and includecom

The files `includepar` and `includecom` are the two most important source code files of FLEXPART . File `includepar` contains all relevant parameter settings, both regarding physical constants as well as field dimensions. File `includecom` defines all FLEXPART variables and fields of a global nature, i.e., those variables and fields shared by all subroutines. While FLEXPART can be run without knowing much about programming and without changing the settings, it is important that the field dimensions specified in `includepar` are compatible with (i.e., as large or larger than) the dimensions of the meteorological input wind fields, the settings specified for the number of particles to be used, output field dimensions, etc. As many different combinations are possible and the memory required by FLEXPART is determined by the various field dimensions, it is recommended that they are adjusted to actual needs in file `includepar` in order to run FLEXPART efficiently and minimize the memory requirements.

3.2 Input data organisation

Input data are organised such that all input data of a "mother domain" valid for one date must be contained in a single file, and all the files for the different dates must be stored in the same directory, which is specified in the `pathnames` file in FLEXPART 's working directory. In addition, a file (we use `AVAILABLE`, but it may have a different name) must be provided, too (`filename` also given in `pathnames`), which contains a list of all available files (their file names and validation dates). The maximum dimensions of these fields are specified by the parameters `nxmax, nymax, nuvzmax, nwzmax, nzmax` in file `includepar`, for x, y, and three z dimensions, respectively. The three z dimensions are for the original ECMWF data (`nuvzmax, nwzmax`) and transformed data (see below) (`nzmax`), respectively. Note that for efficient memory usage, it is important to set these dimensions as small as possible, given the input data to be used. Grid dimensions and other basic things are checked in routine `gridcheck.f`, and error messages are issued if the settings in file `includepar` are too small to accomodate the input grid.

The longitude/latitude region specified by the input grid is also used as the computational domain, i.e., all internal FLEXPART coordinates run from the leftmost/lowermost domain boundary with coordinate (0,0) to the rightmost/uppermost domain boundary with coordinate (nx-1,ny-1), (nx,ny) being the actual input grid dimensions. If global (in longitude direction) input data are used, FLEXPART repeats the westernmost grid cells at the easternmost domain "boundary", in order to facilitate interpolation on all locations of the globe (e.g., if input data run from 0 to 359 ° with 1 ° resolution, data for 0 ° are repeated at 360 °). Particles leaving the grid on one side are moved to the other end of the grid. From FLEXPART version 5.0, the mother domain can be shifted by `nxshift` (file `includepar`) data columns (subroutines `shift_field.f` and `shift_field_0.f`). This allows FLEXPART to produce output for limited domains whose grid overlaps the original domain "boundaries" of the global input data. For instance, a limited domain stretching from 320 ° to 30 ° to be nested in the mother grid of the above example could be accomodated by shifting the mother grid by 30 °. This option is also useful if nested grids (see below) overlap the domain "boundaries".

The pole represents a singularity in a latitude/longitude grid. Thus, horizontal winds (variable fields `uu,vv`) poleward of latitudes `switchnorth, switchsouth` (file `includepar`) are transformed to a polar stereographic projection (variable fields `uupol,vvpol`, which are also stored on the latitude/longitude grid). Particle coordinates poleward of latitudes `switchnorth, switchsouth` are transformed to the polar stereographic projection, too, and particles are then advected on the polar stereographic projection using (`uupol,vvpol`)

taken from the latitude/longitude grid. Note that, as the wind information is kept on the latitude/longitude grid, no interpolation error is made.

Nesting is facilitated since FLEXPART version 3.1. The pole must not be contained in a nest, though. Any number of nesting levels can be used up to a maximum specified by the parameter `maxnests` in file `includepar`. The input files for each nesting level must be stored in a different directory, as specified in the `pathnames` file in a list following the information for the mother domain. The order of this list specifies the order of the nesting levels. Given a certain particle position, the last level is checked first whether it contains the particle or not. If not, the next nesting level is checked, and so forth until the mother domain is reached. For each nesting level, an `AVAILABLE` file must be specified, too, in `pathnames`, which contains the appropriate filenames. All these `AVAILABLE` files must contain the same dates as the one for the mother domain. The horizontal field dimensions of the nests can differ from each other, but must be less than `nxmaxn, nymaxn` set in file `includepar`. As the memory occupied by FLEXPART is determined by `nxmaxn, nymaxn`, which must accomodate the largest nested field, it is most economical to have nests with comparable dimensions. The vertical discretization (see below) must be identical (i.e., no nesting in the vertical direction).

3.3 Vertical model structure and required data

FLEXPART needs five three-dimensional fields: horizontal and vertical wind components, temperature and specific humidity. It expects the input data on ECMWF model (i.e. η) levels which are defined by a hybrid coordinate system. The conversion from η to pressure coordinates is given by $p_k = A_k + B_k p_s$ and the heights of the η surfaces are defined by $\eta_k = A_k/p_0 + B_k$, where η_k is the value of η at the k^{th} model level, p_s is the surface pressure and p_0 is a pressure constant (101325 Pa). A_k and B_k are coefficients, chosen in such a way that the levels closest to the ground follow the topography, while the highest levels coincide with pressure surfaces. Intermediate levels show a gradual transition between topography-following and pressure levels. The vertical wind in hybrid coordinates is not directly available from the ECMWF archives (it is available only in pressure coordinates (i.e., as ω), even on η surfaces), but is calculated from spectral data by the pre-processor that prepares the FLEXPART input data. A surface level is defined in addition to the regular ECMWF model η levels. 2 m temperature, 10 m winds and specific humidity from the first regular model level are assigned to this level, to represent "surface" values. The inaccuracy introduced by the non-vanishing wind speed at the surface is marginal in

most situations.

Mixing heights and parameterized random velocities (see sections 4 and 5) are given in meters and meters per second, respectively, and not in η coordinates. Therefore, in order to avoid time-consuming coordinate transformations during the trajectory computations, all three-dimensional data are transformed from the ECMWF model levels to terrain-following Cartesian z coordinates. The heights of the z levels are kept constant throughout the simulation. They are determined by choosing a grid point with low topography from the first available input field. At this grid point (and at this time), the z levels equal exactly the heights of the η levels. For other grid points and other times, z levels and η levels differ. But choosing a grid point with little topography minimizes the differences, at least close to the ground and for areas with low topography, and this reduces interpolation errors. All data are interpolated to the z levels using linear interpolation (subroutines `verttransform.f` and `verttransform_nests.f`).

Surface parameters are needed as two-dimensional fields: surface pressure, total cloud cover, 10 m horizontal wind components, 2 m temperature and dew point temperature, large scale and convective precipitation, sensible heat flux, solar radiation, and east/west and north/south surface stress. If surface stress and heat flux data are not available, friction velocity and surface sensible heat flux are computed from 10 m wind, 2 m temperature and their respective values at the first model level using the profile method (Berkowicz and Prahm, 1982). All other parameters must be available.

Topography, land-sea-mask and subgrid standard deviation of topography must be contained in the ECMWF input data, too. If dry deposition of gases is to be calculated, a landuse inventory must be provided in an extra file (`landuse.asc`). For Europe, the inventory of Velde *et al.* (1994) is used as a default, but for other regions the user must change the input accordingly.

Chapter 4

Physical parameterization of boundary layer parameters

Accumulated heat fluxes and surface stresses are available for forecasts at the ECMWF. The pre-processor used to extract the input data from the ECMWF archives (provided independently from the FLEXPART source code) selects the appropriate short-term (3 h and 6 h) forecasts and deaccumulates the accumulated flux data. Given that deaccumulated flux data are available, friction velocities u_* and surface sensible heat fluxes $(\overline{w'\Theta'_v})_0$ are calculated from the deaccumulated surface stresses and heat fluxes (Wotawa *et al.,* 1996). The total surface stress is computed from

$$\tau = \sqrt{\tau_1^2 + \tau_2^2}\,, \tag{4.1}$$

where τ_1 and τ_2 are the surface stresses in east/west and north/south direction, respectively. Friction velocity is then

$$u_* = \sqrt{\tau/\rho}\,, \tag{4.2}$$

where ρ is the air density. These calculations are done in subroutine `scalev.f`.

If deaccumulated surface stresses and surface sensible heat fluxes are not available, the profile method after Berkowicz and Prahm (1982) (subroutine `pbl_profile.f`) is applied to wind and temperature data provided at the second model level and at 10 m (for wind) and 2 m (for temperature) (note that previously the first model level was used, but as ECMWF has its first model level now close to 10 m, the second level is used instead). This is an iterative procedure. For its initialization, Obukhov length L is set to a large value. The following three equations are solved in five iterations which is sufficient to

achieve convergence in (almost) all situations:

$$u_* = \frac{\kappa \Delta u}{\ln \frac{z_l}{10} - \Psi_m(\frac{z_l}{L}) + \Psi_m(\frac{10}{L})} \,, \tag{4.3}$$

$$\Theta_* = \frac{\kappa \Delta \Theta}{R\left[\ln \frac{z_l}{2} - \Psi_h(\frac{z_l}{L}) + \Psi_h(\frac{2}{L})\right]} \,, \tag{4.4}$$

$$L = \frac{\overline{T} u_*^2}{g \kappa \Theta_*} \,, \tag{4.5}$$

where κ is the von Kármán constant (0.4), z_l is the height of the first model level, Δu is the difference between wind speed at the first model level and at 10 m, $\Delta\Theta$ is the difference between potential temperature at the first model level and at 2 m, Ψ_m and Ψ_h are the stability correction functions for momentum and heat (e.g., Businger *et al.,* 1971; Beljaars and Holtslag, 1991), g is the acceleration due to gravity, Θ_* is the temperature scale and $\overline{T}$ is the average surface layer temperature (taken as the temperature at the first model level).

The heat flux is then computed by

$$(\overline{w'\Theta'_v})_0 = -\rho c_p u_* \Theta_* \,, \tag{4.6}$$

where ρc_p is the specific heat capacity of air at constant pressure.

Whenever possible, deaccumulated flux data from ECMWF should be provided since they are consistent with the ECMWF model calculations. Wotawa and Stohl (1997) have shown that friction velocities and heat fluxes calculated using the profile method are much less accurate.

Planetary boundary layer (PBL) heights are calculated according to Vogelezang and Holtslag (1996) using the critical Richardson number concept (subroutine `richardson.f`). The PBL height h_{mix} is set to the height of the first model level l for which the Richardson number

$$Ri_l = \frac{(g/\Theta_{v1})(\Theta_{vl} - \Theta_{v1})(z_l - z_1)}{(u_l - u_1)^2 + (v_l - v_1)^2 + 100 u_*^2} \,, \tag{4.7}$$

exceeds the critical value of 0.25. Θ_{v1} and Θ_{vl} are the virtual potential temperatures, z_1 and z_l are the heights, and (u_1, v_1), and (u_l, v_l) are the wind speed components at the 1st and l^{th} model level, respectively. Equation 4.7 holds only for stable and neutral conditions, but it is extended to convective situations by replacing Θ_{v1} with

$$\Theta'_{v1} = \Theta_{v1} + 8.5 \frac{(\overline{w'\Theta'_v})_0}{w_* c_p} \,, \tag{4.8}$$

where

$$w_* = \left[\frac{(\overline{w'\Theta'_v})_0 g h_{mix}}{\Theta_{v1} c_p} \right]^{1/3} \tag{4.9}$$

is the convective velocity scale. The second term on the right hand side of equation 4.8 represents a temperature excess of rising thermals. As w_* is unknown beforehand, h_{mix} and w_* are calculated iteratively for unstable conditions.

Spatial and temporal variations of PBL heights on scales not resolved by the ECMWF model play an important role in determining the thickness of the layer over which the tracer material is effectively distributed. To elucidate this, we first consider unresolved temporal variations. Given a typical evolution of a convective mixed layer, the PBL height reaches its maximum value (say 1500 m) in the afternoon (for instance at 1700 LST), before a much shallower stable PBL forms. Now, if the meteorological data are available only at 1200 LST and at 1800 LST and the PBL heights at those times are, say, 1200 m and 200 m, and some standard interpolation procedure is used to determine the PBL height at 1700 LST, the PBL height is significantly underestimated. In the above example, using linear interpolation, a value of 370 m would have been obtained instead of 1500 m. If a tracer release takes place shortly before the breakdown of the convective mixed layer, this leads to a serious overestimation of the surface concentrations (a factor of four in the above example). Even if the release occurs already in the morning hours, the thickness of the tracer cloud in the evening would be restricted to 1200 m in the above example, whereas the correct thickness would be 1500 m.

Similar arguments hold for spatial variations of PBL heights. A major reason for spatial variability of PBL heights is complex topography. It increases mechanical mixing due to enhanced shear stress, induces gravity waves during stable conditions, and it provides elevated sources of sensible heat during daytime. All these complex effects can produce turbulence and can thereby locally increase the PBL heights above the elevated areas. In addition to topographical effects, significant variations of the PBL heights can also be caused by differences in landuse or soil wetness (Hubbe *et al.,* 1997). If a tracer cloud travels over such a patchy surface, its thickness would be determined by the maximum PBL height experienced along its path rather than by the average PBL height, which is obtained at the grid points of a coarse resolution meteorological model.

The best solution to this problem obviously is to use meteorological data with higher space and time resolution than the data from ECMWF. Another solution would be to use a simple prognostic boundary layer model as an interpolation tool. However, for the current FLEXPART version, we used a somewhat arbitrary parameterization to avoid a significant bias in the tracer cloud thickness which would result in strong overestimations of the

surface tracer concentrations. To account for spatial variations induced by topography, we use an "envelope" PBL height

$$H_{env} = h_{mix} + \min\left[\sigma_Z, \max\left(c\frac{V}{N}, 0\right)\right] \ . \tag{4.10}$$

Here, σ_Z is the ECMWF model subgrid topography, c is a constant (here: 2.0), V is wind speed, N is the Brunt-Vaisala frequency, and $\frac{V}{N}$ is the local Froude number. H_{env} rather than h_{mix} is used for all subsequent calculations. In addition, to account for temporal and other spatial PBL height variations, we do not interpolate H_{env} to the particle position, but take the maximum H_{env} of the eight grid points surrounding a particle's position in space and time. This parameterization removes the overestimation of the surface concentrations which is otherwise found, and also slightly improves the simulation of the horizontal transport patterns.

Chapter 5

Particle transport and diffusion

5.1 Particle trajectory calculations

FLEXPART has been developed on the basis of FLEXTRA (Stohl *et al.,* 1995), a kinematic trajectory model. In order to save CPU time, FLEXPART , contrary to FLEXTRA, generally uses the simple 'zero acceleration' scheme

$$\mathbf{X}(t + \Delta t) = \mathbf{X}(t) + \mathbf{v}(\mathbf{X}, t)\Delta t\,, \tag{5.1}$$

which is accurate to the first order, to integrate the trajectory equation

$$\frac{d\mathbf{X}}{dt} = \mathbf{v}[\mathbf{X}(t)]\,, \tag{5.2}$$

with t being time, Δt the time increment used by the model, $\mathbf{X}$ the position vector, and $\mathbf{v} = \overline{\mathbf{v}} + \mathbf{v}_t + \mathbf{v}_m$ the wind vector that is composed of the grid scale wind $\overline{\mathbf{v}}$, the turbulent wind fluctuations $\mathbf{v}_t$ (Zannetti, 1992) and the mesoscale wind fluctuations $\mathbf{v}_m$. Δt is flexible and determined from criteria discussed in section 5.3. With sufficiently small time steps, the approximation 5.1 yields accurate trajectories.

With FLEXPART version 5.0, numerical accuracy has been improved by making one iteration of the Petterssen scheme (which is accurate to the second order) whenever this is possible, but only for the grid-scale winds. It is implemented as a correction applied to the position obtained with the 'zero acceleration' scheme. In three cases it cannot be applied. First, the Petterssen scheme needs winds at two times, the latter of which may lie outside the two wind fields kept in memory. Second, if a particle crosses the boundaries of nested domains, and third in the PBL if `ctl`> 0.

Both particle transport and turbulent dispersion are handled by the subroutine `advance.f`. In this subroutine all Langevin equations (see below) are solved, and calls are issued to

several interpolation procedures that interpolate winds, surface fields required for parameterizations, etc.

5.2 The Langevin equation

Wind speeds are given on a grid with a resolution that is much too coarse to represent the turbulent eddies occurring in the PBL. These turbulent motions $\mathbf{v}_t$ therefore have to be parameterized. This is done assuming a Markov process based on the Langevin equation (Thomson, 1987) for the individual wind components i

$$dv_{ti} = a_i(\mathbf{x}, \mathbf{v}_t, t)dt + b_{ij}(\mathbf{x}, \mathbf{v}_t, t)dW_j \,, \tag{5.3}$$

where the drift term a and the diffusion term b are functions of the position, the turbulent velocity and time. dW_j are incremental components of a Wiener process with mean zero and variance dt, which are uncorrelated with the other components and are uncorrelated in time. This form of the Langevin equation was described by Legg and Raupach (1982).

Gaussian turbulence is assumed in FLEXPART , which is strictly valid only during stable and neutral conditions. During convective condtions, when turbulence is skewed (i.e. larger areas are occupied by downdrafts than by updrafts), this assumption is violated, but for long-range transport the error due to this is not so large. Cross-correlations between the different wind components are not taken into account, since Uliasz (1994) has shown that they have little effect for long-range dispersion.

Although density decreases with height, most particle models assume constant density flows. Especially for deep boundary layers, this is not justified. Stohl and Thomson (1999) have developed a density correction for Gaussian turbulence that takes the density effect into account.

With the above assumptions, the Langevin equation for the vertical wind component can be written as

$$dw = -w\frac{dt}{\tau_{L_w}} + \frac{\partial \sigma_w^2}{\partial z}dt + \frac{\sigma_w^2}{\rho}\frac{\partial \rho}{\partial z}dt + \left(\frac{2}{\tau_{L_w}}\right)^{1/2} \sigma_w dW \,, \tag{5.4}$$

where w is the turbulent vertical wind component, σ_w is the standard deviation of the turbulent vertical wind component, τ_{L_w} is the Lagrangian timescale for the vertical velocity autocorrelation and ρ is density. The second and the third term on the right hand side are the drift correction (e.g. McNider *et al.,* 1988) and the density correction (Stohl and Thomson, 1999), respectively. This Langevin equation is identical (except for the density correction term) to the one described by Legg and Raupach (1982).

Alternatively, the Langevin equation can be re-expressed in terms of w/σ_w instead of w (Wilson *et al.,* 1983):

$$d\left(\frac{w}{\sigma_w}\right) = -\frac{w}{\sigma_w}\frac{dt}{\tau_{L_w}} + \frac{\partial\sigma_w}{\partial z}dt + \frac{\sigma_w}{\rho}\frac{\partial\rho}{\partial z}dt + \left(\frac{2}{\tau_{L_w}}\right)^{1/2} dW \ , \tag{5.5}$$

This form was shown by Thomson (1987) to fulfill the well-mixed criterion which states that "if a species of passive marked particles is initially mixed uniformly in position and velocity space in a turbulent flow, it will stay that way" (Rodean, 1996). Although the method proposed by Legg and Raupach (1982) violates this criterion in strongly inhomogeneous turbulence, their formulation was found to be practical, as numerical experiments have shown that it is numerically more robust against an increase in the integration time step used by the model. Therefore, equation 5.4 is used when long time steps are allowed to save computation time (see section 5.3); otherwise, equation 5.5 is used. For the horizontal wind components, the Langevin equation is identical to equation 5.4, with no drift and density correction terms.

For the discrete time step implementation of the above Langevin equations (at the example of equation 5.5), two different methods are used. When $(\Delta t/\tau_{L_w}) < 0.5$,

$$\left(\frac{w}{\sigma_w}\right)_{k+1} = \left(1 - \frac{\Delta t}{\tau_{L_w}}\right)\left(\frac{w}{\sigma_w}\right)_k + \frac{\partial\sigma_w}{\partial z}\Delta t + \frac{\sigma_w}{\rho}\frac{\partial\rho}{\partial z}\Delta t + \left(\frac{2\Delta t}{\tau_{L_w}}\right)^{1/2}\zeta \ , \tag{5.6}$$

where ζ is a normally distributed random number with mean zero and unit standard deviation. The subscripts k and $k+1$ refer to subsequent times separated by Δt. When $(\Delta t/\tau_{L_w}) \geq 0.5$,

$$\left(\frac{w}{\sigma_w}\right)_{k+1} = r_w\left(\frac{w}{\sigma_w}\right)_k + \frac{\partial\sigma_w}{\partial z}\tau_{L_w}\left(1 - r_w\right) + \frac{\sigma_w}{\rho}\frac{\partial\rho}{\partial z}\tau_{L_w}\left(1 - r_w\right) + \left(1 - r_w^2\right)^{1/2}\zeta \ , \tag{5.7}$$

where $r_w = \exp(-\Delta t/\tau_{L_w})$ is the autocorrelation of the vertical wind.

If a particle reaches the surface or the top of the PBL, it is reflected and the sign of the turbulent velocity is changed (Wilson and Flesch, 1993). If a particle reaches the top level of the model, it is set back to a position just underneath the highest level. If a particle reaches the lateral boundary of the model domain, the computation of its trajectory is terminated.

5.3 Determination of the time step

$\mathrm{F_L e X^p_a R_T}$ can be used with one of two different options. The faster one (`ctl`<0 in file `COMMAND`) does not adapt the computation time step to the Lagrangian timescale for

the autocorrelations of the turbulent velocity fluctuations. With this option, FLEXPART uses constant time steps of one synchronisation time interval (`lsynctime`, specified in file `COMMAND`, typically 900 seconds). Usually, with reasonably large `lsynctime`, autocorrelations will be very low in this mode. Hence, turbulence is not described well and the density correction does also not work properly. Multiple reflections of the particle at the surface and the PBL top may occur. Nevertheless, for large scale applications, FLEXPART works very well with this option (Stohl *et al.,* 1998), although surface concentrations are usually somewhat underestimated (a few percent, typically, but depending on the PBL conditions).

If the transport in the PBL shall be described more accurately, the time steps must be limited by the Lagrangian time scales. Since the vertical wind component is the most important one, only τ_{L_w} is used to limit the time step. The user must specify two constants, c_{tl} and $ifine$ in file `COMMAND`. The first one determines the time step Δt_i according to

$$\Delta t_i = \frac{1}{c_{tl}} \min \left(\tau_{L_w} \, , \, \frac{h}{2w} \, , \, \frac{0.5\sigma_w}{\partial \sigma_w / \partial z} \right) \, . \tag{5.8}$$

The minimum value of Δt_i is 1 second. Δt_i is used for solving the Langevin equations for the horizontal turbulent wind components.

For solving the Langevin equation for the vertical wind component, a shorter time step $\Delta t_w = \Delta t_i / ifine$ is used. However, note that there is no interaction between horizontal and vertical wind components on timescales less than Δt_i. This strategy (given sufficiently large values for c_{tl} and $ifine$) ensures that the particles stay vertically well-mixed also in very inhomogeneous turbulence, while keeping the computational cost at a minimum (i.e., solution of the Langevin equation for w only at the shortest time steps Δt_w).

5.4 The parameterization of the wind fluctuations

The solution of the Langevin equations requires the knowledge of $\sigma_{v_t i}$ and τ_{L_i} at any time and at any position of a particle trajectory. For their determination, Hanna (1982) proposed a parameterization scheme that is based on the boundary layer parameters h, L, w_*, z_0 and u_*, i.e. PBL height, Monin-Obukhov length, convective velocity scale, roughness length and friction velocity, respectively. It is used here (subroutines `hanna.f, hanna1.f, hanna_sho`) with a modification taken from Ryall and Maryon (1997) for σ_w for convective conditions, as Hanna's scheme does not always yield smooth profiles of σ_w throughout the whole PBL, leading to an unmixing of well-mixed particles. In the following, the subscript u refers to the along-wind components, v to the cross-wind components and w to the vertical

components of the turbulent velocities; f is the Coriolis parameter.

Unstable conditions:

$$\frac{\sigma_u}{u_*} = \frac{\sigma_v}{u_*} = \left(12 + \frac{h}{2|L|}\right)^{1/3} \tag{5.9}$$

$$T_{L_u} = T_{L_v} = 0.15\frac{h}{\sigma_u} \tag{5.10}$$

$$\frac{\sigma_w}{w_*} = \left[1.2\left(1 - 0.9\frac{z}{h}\right)\left(\frac{z}{h}\right)^{2/3} + \left(1.8 - 1.4\frac{z}{h}\right)u_*^2\right]^{1/2} \tag{5.11}$$

For $z/h < 0.1$ and $z - z_0 > -L$:

$$T_{L_w} = 0.1\frac{z}{\sigma_w\left[0.55 - 0.38\left(z - z_0\right)/L\right]} \tag{5.12}$$

For $z/h < 0.1$ and $z - z_0 < -L$:

$$T_{L_w} = 0.59\frac{z}{\sigma_w} \tag{5.13}$$

For $z/h > 0.1$:

$$T_{L_w} = 0.15\frac{h}{\sigma_w}\left[1 - \exp\left(\frac{-5z}{h}\right)\right] \tag{5.14}$$

Neutral conditions:

$$\frac{\sigma_u}{u_*} = 2.0\exp(-3fz/u_*) \tag{5.15}$$

$$\frac{\sigma_v}{u_*} = \frac{\sigma_w}{u_*} = 1.3\exp(-2fz/u_*) \tag{5.16}$$

$$T_{L_u} = T_{L_v} = T_{L_w} = \frac{0.5z/\sigma_w}{1 + 15fz/u_*} \tag{5.17}$$

Stable conditions:

$$\frac{\sigma_u}{u_*} = 2.0\left(1 - \frac{z}{h}\right) \tag{5.18}$$

$$\frac{\sigma_v}{u_*} = \frac{\sigma_w}{u_*} = 1.3\left(1 - \frac{z}{h}\right) \tag{5.19}$$

$$T_{L_u} = 0.15\frac{h}{\sigma_u}\left(\frac{z}{h}\right)^{0.5} \tag{5.20}$$

$$T_{L_v} = 0.07\frac{h}{\sigma_v}\left(\frac{z}{h}\right)^{0.5} \tag{5.21}$$

$$T_{L_w} = 0.1 \frac{h}{\sigma_w} \left(\frac{z}{h} \right)^{0.5} \tag{5.22}$$

The minimum τ_{L_u}, τ_{L_v} and τ_{L_w} used are 10 seconds, 10 seconds and 30 seconds, respectively, in order to avoid excessive computation times for particles close to the surface. Along-wind and cross-wind components are transformed to Cartesian wind components.

Above the PBL ($z > h$), version 6.0 uses a revised scheme. In the stratosphere, a vertical diffusivity of 0.1 $\mathrm{m^2 s^{-1}}$ is used, following recent work of Legras et al. (2003), whereas in the troposphere a diffusivity of 50 $\mathrm{m^2 s^{-1}}$ for the horizontal wind components is used. Stratosphere and troposphere are distinguished based on a threshold of 2 pvu (potential vorticity units), but altitudes below 5 km are always assumed to be tropospheric and altitudes above 18 km are always assumed to be stratospheric.

5.5 Mesoscale velocity fluctuations

Mesoscale motions are neither resolved by the ECMWF data nor covered by the turbulence parameterization which is based on measurements that are representative for a temporal scale of less than an hour and correspondingly short length scales. This spectral gap must be filled, since mesoscale motions can significantly accelerate the growth of a dispersing plume (Gupta *et al.,* 1997). For this, we use a method that is similar to the one recently described by Maryon (1998), namely to solve an independent Langevin equation for the mesoscale wind velocity fluctuations ("meandering" in Maryon's terms), assuming that they are largely independent from the turbulent fluctuations covered by Hanna's (1982) parameterization. This is done in `advance.f` using time steps of `lsynctime` only. In contrast to Maryon (1998), who used time scales and velocity variances derived from a spectral analysis of a time series of wind measurements at a single station, we assume that the variance of the wind observed at the grid scale provides some information on its subgrid variance. The wind velocity standard deviation used for the mesoscale Langevin equation is set to `turbmesoscale` (set in file `includepar`) times the standard deviation of the 16 grid points surrounding the particles' position in space and time. The corresponding time scale is taken as half the interval at which wind fields are available, assuming that the linear interpolation between the grid points can recover half the subgrid variability. According to Stohl *et al.* (1995), who compared wind field data at different resolution, this is not an unlikely assumption. This empirical approach does not describe actual mesoscale phenomena, but it is similar to the ensemble methods which are useful to assess trajectory accuracy (Kahl, 1996; Baumann and Stohl, 1997; Stohl, 1998). It provides an estimate of how the grid scale dispersion reacts to perturbations on smaller scales. It is not so

important in the PBL, where the interaction of the vertical shear of the mean horizontal wind and the turbulent vertical motions dominates plume dispersion, but it is significant for long-range transport in the free troposphere.

The factor `turbmesoscale` used to scale mesoscale velocity fluctuations has a subjective value, "tuned" to yield realistically broad plumes in the free troposphere. It appeared that pollution plumes during intercontinental transport episodes in the free troposphere were too broad in FLEXPART versions 4.x compared to satellite images (e.g., clouds in Meteosat, TOMS aerosol index), possibly related to the introduction of the convection scheme with version 4.0, which also tends to broaden the plumes. With FLEXPART version 5.0, therefore, this scale has been reduced by a factor of 3 compared to previous FLEXPART versions. It is now set to 0.16. Further reduction results in too narrow plumes and is not recommended.

5.6 Convection

5.6.1 The significance of convection

Vertical transports occur not only on scales resolved by global models such as the ECMWF forecast model. An important transport mechanism are the updrafts in convective clouds. They occur in conjunction with downdrafts within the clouds and compensating subsidence in the cloud-free surroundings. These convective transports are transports that are grid-scale in the vertical, but sub-grid scale in the horizontal. This means that they are not represented by the vertical velocity delivered by large-scale models, and that they cannot be properly represented just by (enhanced) turbulence. Convective transport is an essential element in the tropical Hadley circulation and it is necessary to represent it in models of global tracer transport. Convective transport can be important in specific synoptic systems also outside the tropics.

5.6.2 General aspects of the scheme

To represent convective transport in a particle dispersion model, it is necessary to know how the particles shall be vertically redistributed. This means the destination level of each particle must be determined (which is not necessarily the level just above or below the original level). For FLEXPART we chose the convective parameterization scheme by *Emanuel and Živković-Rothman* [1999], as it relies on the grid-scale temperature and humidity fields and gives as a by-product a displacement matrix providing the necessary

information for the particle redistribution.

The implementation of a convective parameterisation scheme in an off-line model is not trivial, even more so as we have no influence on the input data provided [in the case of ECMWF data]. The present scheme was coupled with FLEXPART for use with windfields from either ECMWF (ECMWF, 1995) or NCEP GFS (National Center for Environmental Prediction Global Forecast System). Other data sets may contain specific, convection-related fields and the present procedure may be updated in the future to make use of such information.

Convection adds a significant burden to the computation time. It scales to the square of the number of vertical model levels, and with the current 60-level ECMWF fields, may account for up to 70% of FLEXPART 's computation time, which still is significantly less than in the first implementation by *Seibert et al.* [2001].

5.6.3 Detailed description

The convection is computed within the subroutines `convmix.f`, `calcmatrix.f`, `convect43c.f`, and `redist.f`, which are described in more detail below. Subroutine `convmix.f` is the interface between FLEXPART and the convection scheme and handles the subroutine calls related to convective mixing. In `calcmatrix.f` different variables are initialized for the call of `convect43c.f`, where the convective drafts and the redistribution matrix are determined. Finally, the particles are redistributed in `redist.f`.

In the Emanuel scheme (subroutine `convect43c.f`), convection in a grid cell is triggered whenever the virtual temperature of a parcel lifted to the first level above its condensation level exceeds the environmental virtual temperature at this level by more than a critical threshold. Based on the buoyancy sorting principle, which is decribed in more detail in *Emanuel [1991]*, *Raymond and Blyth* [1986], and *Telford* [1975], saturated upward and downward mass fluxes within clouds and unsaturated precipitating downward mass fluxes outside of the clouds are calculated by accounting for entrainment and detrainment. The scheme then provides the tendencies for the resolved-scale temperature and humidity and a displacement matrix containing the mass fractions that are displaced by the saturated convective drafts from each level i to each other level j. We then calculate the saturated up- and downdrafts from the matrix and assume that these saturated mass fluxes are balanced by a subsidence mass flux in the environment.

The convection scheme is called every FLEXPART time step which is usually 900 s. The temperature (T) and specific humiditiy (Q) profiles are read in from ECMWF (NCEP GFS) data every 3 hours (which is typically the temporal resolution of the ECMWF

(NCEP GFS) data) and are interpolated to the current time every $\mathrm{F_LEXP_aRT}$ time step.

For efficiency reasons, particles are sorted according to their horizontal grid positions (subroutine `sort2.f` in `convmix.f`) before they are redistributed. This enables us to proceed particle-wise instead of grid-cell wise, with the benefit that the convection subroutines (`calcmatrix.f` and `convect43c.f`) are called only for grid cells actually containing particles and only once for each such cell. All other options would require prohibitive amounts either of memory or of computational work. The particles in each convectively active box are then redistributed (`redist.f`) according to the displacement matrix provided by the convection scheme. The vertical grid position of the particle before the convection is determined and a random number between [0,1] is chosen. The mass fluxes from a level i that are displaced to all the other levels j are divided through the mass contained in level i in order to obtain the probabilities by which a particle is randomly (depending on the random number) displaced from level i to another level j. After the convective redistribution of the particles, the compensating subsidence is compiled to a vertical velocity and acts on those particles in the grid box that are not displaced by convective drafts. This accounts for the overall effect of convection on the environment.

5.7 Particle splitting

During the initial phase of dispersion from a point source in the atmosphere, the particles stay relatively close together and form a compact cloud. Relatively few particles suffice to simulate this initial phase correctly. After some time, however, the particle cloud gets distorted and particles spread over a much larger area. More particles are now needed. To save computing time for short travel times, but nevertheless give reliable results for longer travel times, $\mathrm{F_LEXP_aRT}$ allows the user to specify a time constant Δt_s (file `COMMAND`). Particles are split into two (each of which receives half of the mass of the original particle) after travel times of Δt_s, $2\Delta t_s$, $4\Delta t_s$, $8\Delta t_s$, and so on (subroutine `timemanager.f`).

5.8 Backward modelling

Backward modelling with $\mathrm{F_LEXP_aRT}$ can be used for the calculation of source-receptor matrices. The backward mode is more efficient than the forward mode if the number of receptors is smaller than the number of (potential) sources. In this case, either mass mixing ratios or concentrations (this can be selected in the `COMMAND` file using the variable `indicator_backward`) are the quantity underlying the $\mathrm{F_LEXP_aRT}$ calculations and which

have to be specified in FLEXPART 's `RELEASES` file instead of a release mass. The gridded output of FLEXPART for a backward run (output is into files `grid_time_date`, where `date` is an identifier for both date and time) is then a response function ("sensitivity") to emission input. It is proportional to the residence time in the output grid cell (during the output averaging time interval) of the air arriving at the receptor. Since FLEXPART version 5.0, density variations in the atmosphere are correctly taken into account, leading to a unit of the output response function of s $m^3 kg^{-1}$ if mixing ratio input was selected. If folded with a 3-d field of emission fluxes into the output grid boxes (in kg m^{-3}), a mixing ratio is obtained, if the input was selected to be mixing ratio. In the case of loss processes (dry or wet deposition, decay) the response function is "corrected" for these loss processes. For more information on the theory of backward modelling, see Seibert (2001) and, particularly, Seibert and Frank (2004). An application illustrating some of the possibilities is described in Stohl et al. (2003).

For backward simulations with FLEXPART , it is important to relate every output field exactly to a certain receptor (remember that receptors are specified in the `RELEASES` file for backward runs). Therefore, the output fields must contain a dimension `maxpoint`, the maximum number of receptor points that can be specified in a `RELEASES` file. This dimension is not needed for the output fields in forward runs. In order to avoid creating a further dimension for the output fields (thus increasing the memory demands of FLEXPART), we replace `maxspec` (used for forward runs) with `maxpoint` (in backward runs) as a dimension of the output fields. This has the disadvantage that only one species can be calculated in a single backward run. To switch from forward to backward runs, the parameter `maxpointspec` is used. It must be set (in the file `includecom`) to `maxspec` for forward runs and to `maxpoint` for backward runs.

5.9 Plume trajectories

In a recent paper, Stohl et al. (2002) propose a method to condense the complex and large FLEXPART output using cluster analysis. The idea behind this is to cluster, at every output time, the positions of all particles belonging to a release (or receptor) point, and write out only clustered particle positions, along with additional information (e.g., how many particles are inside the PBL, or in the stratosphere). This creates information that is similar to traditional trajectories plus their uncertainty trajectories but includes turbulence and convection.

This option can be activated by setting `iout` to 4 or 5 in file `COMMAND`. The number of

clusters is set to a default value of 5, which can be changed with the parameter `ncluster` in file `includepar`. The clustering is handled and output is produced by subroutine `plumetraj.f`. Subroutine `clustering.f` does the clustering, using the iterative algorithm described by Dorling et al. (1992) (except that clustering is done not for entire trajectories at once, but at every time step independently).

Chapter 6

Radioactive decay

Radioactive decay is accounted for by reducing the particle mass at each time step according to

$$M(t + \Delta t) = M(t) \exp(-\Delta t/\beta) \ , \tag{6.1}$$

where M is particle mass, and the time constant $\beta = T_{1/2}/\ln(1/2)$ is determined from the pollutants half life $T_{1/2}$. Deposited pollutant mass decays at the same rate. Half life times must be provided by the user in the `SPECIES` file.

Chapter 7

Wet deposition

Wet deposition removes particles and gases from the atmosphere. In principle, in-cloud and below-cloud scavenging must be separated (Asman, 1995). However, as data on cloud base height and cloud extension are not available, in-cloud and below-cloud scavenging are treated jointly by means of scavenging coefficients. Using scavenging coefficients, wet deposition takes the form of an exponential decay process (McMahon and Denison, 1979)

$$\frac{dM_i}{dt} = -\Lambda_i M_i \ , \tag{7.1}$$

and

$$M_i(t + \Delta t) = M_i(t) \exp(-\Lambda_i \Delta t) \ , \tag{7.2}$$

where M_i and Λ_i are the particle mass and the scavenging coefficient of species i, respectively.

The scavenging coefficients Λ_i are species-dependent and increase with precipitation rate according to

$$\Lambda_i = A_i I^{B_i} \ , \tag{7.3}$$

where I is the precipitation rate in mm/hour, A_i [s^{-1}] is the scavenging coefficient at I =1 mm/hour and B_i gives the dependency on precipitation rate. Both A_i and B_i must be specified by the user in file `SPECIES`. Scavenging coefficients for snow and rain may be different. For the sake of simplicity and for lack of data, however, the same scavenging coefficients are used for both snow and rain.

Precipitation can vary considerably over the area covered by a single grid cell. As wet deposition depends nonlinearly on precipitation rate, this subgrid variability must be accounted for (Hertel *et al.,* 1995). The area fraction which experiences precipitation

Table 7.1: Correction factors used for the calculation of the area fraction that experiences precipitation. Precipitation rates are in mm/hour.

	I_l and I_c				
Factor	$I \leq 1$	$1 < I \leq 3$	$3 < I \leq 8$	$8 < I \leq 20$	$20 < I$
fr_l	0.50	0.65	0.80	0.90	0.95
fr_c	0.40	0.55	0.70	0.80	0.90

given a certain grid-scale precipitation rate is calculated by

$$F = \max\left[0.05, CC\frac{I_l fr_l(I_l) + I_c fr_c(I_c)}{I_l + I_c}\right] , \tag{7.4}$$

where CC is the total cloud cover, I_l and I_c are the large scale and convective precipitation rates, respectively, and fr_l and fr_c are correction factors that depend on I_l and I_c (see table 7.1). The subgrid scale precipitation rate is then

$$I_s = (I_l + I_c)/F . \tag{7.5}$$

To save computing time, wet deposition is not calculated at each integration time step, but only once during each concentration sampling interval. With sampling intervals being shorter than the intervals between available meteorological fields, the inaccuracy introduced by this simplification is marginal.

Chapter 8

Dry deposition

The removal of particles and gases from the atmosphere by turbulent transfer and subsequent uptake or deposition at the surface constitutes a primary means of cleansing the atmosphere. This process, called *dry deposition*, is usually described by a deposition velocity

$$v_d(z) = -F_C/C(z) \ , \tag{8.1}$$

where v_d, F_C and C are the deposition velocity, and the flux and the concentration of a species at height z within the constant flux layer. A constant deposition velocity v_d may be specified to be used by FLEXPART if detailed information is not available for a substance. More comprehensive parameterizations for gases and particles are also available, but these require detailed specifications of the physical and chemical properties of the substance and may introduce additional uncertainties if these are not well known. All information on the deposition characteristics of the tracer considered must be provided by the user in file `SPECIES`.

8.1 Dry deposition of gases

The deposition velocities are calculated with the *resistance method* (Wesely and Hicks, 1977) in subroutine `getvdep.f`. The deposition velocity of gases is given by

$$|v_d(z)| = [r_a(z) + r_b + r_c]^{-1} \ , \tag{8.2}$$

where r_a is the aerodynamic resistance between z and the surface (identical for all gases, but dependent on stability), r_b is the quasilaminar sublayer resistance (dependent on the molecular diffusivity of a species in air and on the thickness of the quasilaminar

sublayer), and r_c is the bulk surface resistance (dependent on the type of surface and on the depositing species).

The aerodynamic resistance r_a can be expressed with the flux-profile relationship which is based on Monin-Obukhov similarity theory (see, e.g., Stull, 1988)

$$r_a(z) = \frac{1}{\kappa u_*}[\ln(z/z_0) - \Psi_h(z/L) + \Psi_h(z_0/L)] \ , \tag{8.3}$$

where κ is the von Kármán constant (0.4), L the Monin-Obukhov length and Ψ_h the similarity function for heat (Businger *et al.,* 1971) which corrects for the influence of stability. r_a is calculated in function `raerod.f`.

Following Erisman *et al.* (1994), the quasilaminar sublayer resistance is

$$r_b = \frac{2}{\kappa u_*}\left(\frac{Sc}{Pr}\right)^{2/3} \ , \tag{8.4}$$

where Sc and Pr are the Schmidt and Prandtl numbers, respectively. Pr is 0.72 and $Sc = \upsilon/D_i$, with υ being the kinematic viscosity of air (typical value 0.15 $\mathrm{cm^2 s^{-1}}$) and D_i being the molecular diffusivity of species i in air. The slight dependency of υ on air temperature is formulated in accordance with Pruppacher and Klett (1978). r_b is calculated in function `getrb.f`.

The surface resistance r_c is given by (Wesely, 1989)

$$r_c = [1/(r_s + r_m) + 1/r_{lu} + 1/(r_{dc} + r_{cl}) + 1/(r_{ac} + r_{gs})]^{-1} \ , \tag{8.5}$$

where r_s, r_m and r_{lu} represent the bulk values for leaf stomatal, leaf mesophyll and leaf cuticle surface resistances (alltogether the upper canopy resistance) , r_{dc} represents the gas-phase transfer affected by buoyant convection in canopies, r_{cl} the resistance of leaves, twig, bark and other exposed surfaces in the lower canopy, r_{ac} the resistance for transfer that depends only on canopy height and density, and r_{gs} the resistance for the soil, leaf litter, etc., at the ground surface. Each of these resistances is parameterized according to chemical reactivity and solubility of the chemical species, and the meteorological conditions. A detailed description of the applied procedure is given by Wesely (1989). r_c is calculated in function `getrc.f`.

The required data on landuse is taken from a European database developed by van de Velde *et al.* (1994). It provides the area fractions of eight landuse classes on a grid with 10' resolution. The roughness lengths z_0 needed for parameterization of r_a are also estimated from this landuse inventory. Table 8.1 shows the landuse classes and associated values of z_0. Charnock's relationship (see Stull, 1988) is used to calculate z_0 for the classes

Table 8.1: List of the landuse classes and roughness lengths used by flexpart.

Grassland	0.10
Arable land	0.15
Permanent crops	0.30
Forest	0.60
Inland water	Equ. 8.6
Urban areas	0.70
Other	0.10
Ocean	Equ. 8.6

"Ocean" and "Inland water", because of its dependence on wave height

$$z_0 = 0.016 u_*^2 / g \ . \tag{8.6}$$

Deposition velocities in a grid cell are calculated for all landuse classes which cover a non-vanishing area of the cell. The average deposition velocity in the cell is then computed by weighting the individual deposition velocities with the respective area fractions of the landuse classes.

Outside the area of the European landuse database, the landuse classes are determined only from the ECMWF land-sea-mask, attributing the landuse classes "Ocean" to the sea surfaces and "Grasslands" to the land surfaces.

8.2 Dry deposition of particulate matter

The deposition of particulates is calculated in subroutine `partdep.f` according to

$$v_d(z) = [r_a(z) + r_b + r_a(z) r_b v_g]^{-1} + v_g \ , \tag{8.7}$$

where v_g is the gravitational settling velocity (Slinn, 1982).

The settling velocity is calculated from

$$v_g = \frac{g \rho_p d_p^2 C_{cun}}{18 \mu} \ , \tag{8.8}$$

where ρ_p and d_p are the particle density and diameter, μ the dynamic viscosity of air (0.000018 kgm^{-1}s^{-1}) and C_{cun} the Cunningham slip-flow correction. The quasilaminar sublayer resistance is calculated from the same relationship as for gases, with an additional impaction term. For further details see Slinn (1982).

Settling and dry deposition velocities are strongly dependent on the particulates sizes. Usually, their sizes cover some range (polydisperse aerosol) and it is not justified to describe particulate matter by a single average diameter. FLEXPART assumes a logarithmic normal distribution of the particulate mass over the range of particulate diameters. The user must specify the mean particulate diameter $\overline{d_p}$ and a measure of the variation around $\overline{d_p}$, σ_p. $\sigma_p = 0.1$ or $\sigma_p = 10$ both indicate that 68% of the mass are between $0.1\overline{d_p}$ and $10\overline{d_p}$. Then, the settling and deposition velocities are calculated for several particle diameters within the range of ± 3 standard deviations of the logarithmic normal distribution. The particulate mass fractions represented by the diameter intervals are used to weight the respective velocities to yield average settling and deposition velocities. A value of $\sigma_p = 1$ indicates monodisperse aerosol, i.e. aerosol with a uniform diameter.

Gravitational settling is important not only for the computation of the dry deposition velocity, but also affects the particles trajectory. For practical reasons, however, gravitational settling velocities are only taken into account for trajectory calculations when the particles represent a single species. If FLEXPART simulates the transport of multiple species, gravitational settling is neglected, because otherwise each species would follow its own trajectory. This would violate the model concept with several species being described by a single particle. Thus, if the dispersion of large particles with significant settling velocities is to be calculated, a single species calculation is suggested.

8.3 Loss of particle mass due to dry deposition

The depositon velocity is calculated for a reference height h_{ref} of 15 m, which is set by parameter `href` in file `includepar`. All particles that are below $2h_{ref}$ are subjected to dry deposition. For all particles below $2h_{ref}$ deposition is described by

$$\frac{dM_i}{dt} = \frac{-v_d M_i}{2h_{ref}} \,, \tag{8.9}$$

The mass lost by deposition is then calculated by

$$\Delta M_i(t) = M_i(t) \left[1 - \exp\left(\frac{-v_d(h_{ref})\Delta t}{2h_{ref}} \right) \right] \,. \tag{8.10}$$

This simple representation allows the deposition velocities to be precalculated on the meteorological grid. A uniform deposition velocity for each species is then applied to all particles between the ground and $2h_{ref}$. This is appropriate especially for the fast option of the model using long time steps. It is less justified when time steps below the

Lagrangian time scale are used, since the deposition process could be described more accurately under these conditions.

Chapter 9

Model output

Tracer concentrations and/or mixing ratios (for forward runs), or emission sensitivity repsonse functions (for backward runs) are calculated on a three-dimensional longitude-latitude grid, defined in file `OUTGRID`, whose domain and resolution may differ from the grid on which meteorological input data are given. Two-dimensional wet and dry deposition fields are calculated over the same spatial domain, and tracer mass fluxes can also be determined on the 3-d grid. Except for the mass fluxes, output can also be produced on a nested output grid, defined in file `OUTGRID_NEST`. The nested output has the same number of output levels as the mother grid, but can have different horizontal resolution. For certain locations, specified in file `RECEPTORS`, concentrations can be calculated independently from this grid (see below).

9.1 Output organisation

9.1.1 Gridded output

The time interval (variable `loutstep`) at which output is produced by $\mathrm{F_L eX^p_a R_T}$ is read in from file `COMMAND`. For every output time, files are created, whose file name ends with the date and time in the format `yyyymmddhhmmss`. A list of all these dates, for which model output is produced, is written to the formatted file `dates`. There are several output options in $\mathrm{F_L eX^p_a R_T}$, which can all be selected in file `COMMAND`. Gridded output fields can be concentrations (output files `grid_conc_date`, unit 10^{-12} kg m^{-3}), mixing ratios (output files `grid_pptv_date`, unit ppt by volume), 'residence times' in backward simulations (uses output option 'concentrations'; output files `grid_time_date`, unit s m^{3}kg^{-1} if mixing ratio is chosen as the backward simulation mode), or fluxes (output

files `grid_flux_date`, unit 10^{-12} kg m^{-2} s^{-1}). Output files `grid_conc*`, `grid_pptv*`, and `grid_time*` also contain wet and dry deposition fields (unit 10^{-12} kg m^{-2}), and all files contain, for each grid cell, corresponding uncertainties (see below how this is determined). All these file types share a common header, file `header` produced by subroutine `writeheader.f`, where important information on the model run (start of simulation, grid domain, number and position of vertical levels, age classes, release points, etc.) is stored. In all postprocessing programs, the header must be read in before the actual data files. File names for the output nests follow the same nomenclature as described above, but with `_nest` added (e.g., `header_nest`, or `grid_conc_nest_date`). The gridded output files are written with subroutines `concoutput.f` and `fluxoutput.f`.

Sparse matrix format

$\mathrm{F_LeX^p_aR_T}$ model output often contains many grid cells with zero concentrations, fluxes, etc. Dumping the grid completely to the output file would create a lot of superfluous output. Therefore, writing out only those cells that actually contain non-zero values may be more effective. In this case, however, the grid indices must be written out together with the concentration values (note that all three grid indices are combined into a single integer number). This makes this method less efficient than a full grid dump if the majority of grid cells contains non-zero concentrations. In order to chose the most effective method, at every output time and for every output field it is determined which option requires less storage space (sparse matrix for mostly zero concentrations, full grid dump for mostly non-zero concentrations), and this option is then used. Refer to the output subroutines in order to know how the output has to be read in with a Fortran program. Note that concentrations for grid cells not written out (in sparse matrix format) have to be initialized with zeroes before the respective read statement.

9.1.2 Receptor point output

For a list of points at the surface, concentrations or mixing ratios can be determined with a more accurate method than for the full grid (see below). This information is written to files `receptor_conc` and `receptor_pptv`, respectively, for all dates of a simulation. This output option cannot be used for backward simulations.

9.1.3 Direct particle dump and warm start option

Particle information (3-d position, release time, release point, and release masses for all species) can also be dumped to files (subroutine `partoutput.f`), either 'only at the end' of a simulation or continuously, like for gridded output, depending on specifications in file `COMMAND`. For continuous output, files `partposit_date` contain the actual data for each output date. For output 'only at the end', particle information is dumped every output interval (as for gridded output) to file `partposit_end`, but this file is overwritten upon each new output. This way, the particle information is kept (almost) actual throughout a simulation. If FLEXPART must be terminated for some reason, it can be continued later on using the information from `partposit_end`.

A warm start is done if variable `ipin` is set to 1 in file `COMMAND`. `readpartpositions.f` first reads file `header` and then file `partposit_end` produced by the previous run in the same directory where the output is to be stored. Note that before such a warm start, `header` and `partposit_end` should be copied and given different filenames, as otherwise they are overwritten upon re-start of FLEXPART and the saved information gets lost if FLEXPART terminates before the next call to `partoutput.f`

If option `mquasilag` (available since version 6.0) is chosen in file `COMMAND`, particle dumps every output interval are produced in a very compact format by converting the positions to an `integer*2` format (subroutine `partoutput_short.f`). As some accuracy is lost in the conversion, this output cannot be used for the warm start option. Another difference to the normal particle output is that, if this option is chosen, every particle gets a unique number (otherwise, they are numbered according to the release location; variable field `npoint`) that allows postprocessing routines to identify continuous particle trajectories.

All output files described so far, except the `dates` file, are in binary format.

9.1.4 Clustered plume trajectories

Condensed particle output using the clustering algorithm described in section 5.9 is written to the formatted file `trajectories.txt`. Information on the release (or receptor for backward runs) points (coordinates, release start and end, number of particles) is written by subroutine `openouttraj.f` to the beginning of file `trajectories.txt`. Subsequently, `plumetraj.f` writes out a time sequence of the clustering results for each receptor/release point: receptor/release point number, time in seconds elapsed since the middle of the release interval, plume centroid position coordinates, various overall statistics (e.g., fraction

of particles residing in the PBL and troposphere), and then for each cluster the cluster centroid position, the fraction of particles belonging to the cluster, and the root-mean-square distance of cluster member particles from the cluster centroid position. In order to construct plume trajectories (i.e., combine the results of cluster analyses for one release/receptor point from subsequent times), `trajectories.txt` must be searched for all entries concerning this particular location.

9.2 Calculation of concentrations and age spectra

9.2.1 Concentrations

The concentration of a pollutant at time t in a grid cell is calculated by sampling the mass fractions of all particles within the grid cell and dividing by the grid cell volume

$$c = \frac{1}{V} \sum_{i=1}^{N} (m_i f_i) \ , \tag{9.1}$$

with V being the grid cell volume, m particle mass, N the total number of particles, and f_i the fraction of the mass of particle i that is attributed to the respective grid cell. This mass fraction is calculated for each particle and each grid cell by a simple uniform kernel with bandwidths $(\Delta x, \Delta y)$, where Δx and Δy are the grid distances on the longitude-latitude output grid. Figure 9.1 illustrates how the mass fractions are calculated. The particle is located at the center of the shaded rectangle with side lengths $(\Delta x, \Delta y)$. Generally, the shaded area stretches over four grid cells. Then, the mass fraction of the particle attributed to each of these four cells equals the fraction of the shaded area that falls within this cell. The uniform kernel is not used during the first 3 hours after a particle's release (when it is attributed only to the grid cell it resides in), in order to avoid artificial smoothing of concentration fields close to the source. The concentration calculations are done in subroutine `conccalc.f`.

9.2.2 Mixing ratios

The user can choose (`iout` in file `COMMAND`) to output mixing ratios instead of (or in addition to) concentrations. Mixing ratios are calculated from concentrations in subroutine `concoutput.f` using air density in the respective grid cell and the molar weight of the respective tracer species. The density of the air is calculated without interpolation from the data on the mother domain.

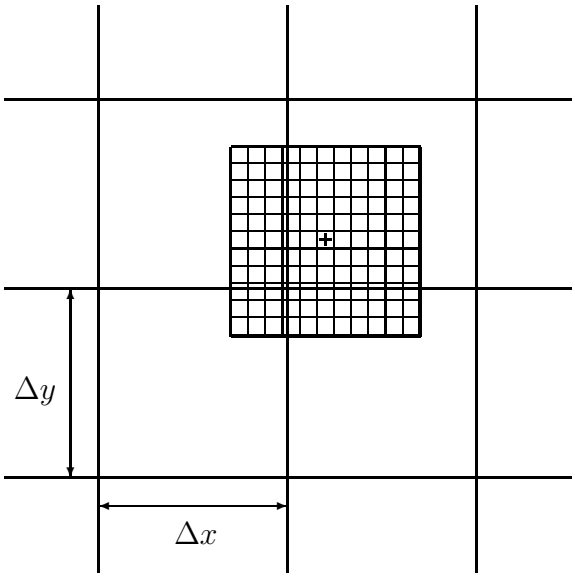

Figure 9.1: Illustration of the uniform kernel used to calculate gridded concentration and deposition fields. The particle position is marked by "+"

9.2.3 Uncertainties

The uncertainty of the output is estimated by carrying `nclassunc` (defined in file `includepar`, typically a value of about 10 suffices) classes of particles in the model simulation, and determining the concentration (or mixing ratio) at every grid point separately for every class (this is done in subroutine `conccalc.f`). Then, at every grid point the concentration's standard deviation is calculated from the individual concentration estimates, which, divided by the square root of the number of classes, yields the standard deviation of the mean concentration (subroutine `concoutput.f`). This number is reported in the output files along with the mean concentration of all particle classes. An overall uncertainty for the whole field is also written periodically to the standard output during the model run. It is reported as relative uncertainty (standard deviation divided by mean concentration). For large uncertainties, a model simulation can thus be stopped at an early stage and re-run with a larger number of particles if required.

Note that the memory needed for some auxiliary fields increases with the number of classes used for the uncertainty calculations AND the number of age classes (see below). It may be necessary to reduce the parameter `nclassunc` in file `includepar` for runs with large output grids and especially with age spectra calculation. It typically must also be reduced (perhaps to 1, i.e., no uncertainty calculations) for the backward mode, if calculations are to be done for a (large) number of release points.

9.2.4 Age spectra

The concentrations can optionally be split into the contributions from particles of different age to create age spectra. The age of a particle is defined as the time passed since its release. The age spectra option is switched on in file `COMMAND` and the age classes are specified in file `AGECLASSES`. Particles expire and are not further tracked once they are older than the oldest age class. The vacant computer storage space, not occupied anymore by these particles, is made available to new particles if required. Therefore, it is sensible to use the age spectra option even with a single age class in order to specify a maximum particle age. Pay attention that the size of the output (and the memory needed for the output fields) increases with the number of age classes.

9.2.5 Parabolic kernel

In addition to the simple uniform kernel method, a computationally much more demanding parabolic kernel method as described in Uliasz (1994) can be used to calculate surface concentrations for a limited number of user-specified receptor points. It should be used for a few receptor sites only, since it can increase the computation time of $\mathrm{FLEXPART}$ considerably. The concentration is estimated from

$$c(x, y, z = 0) = \sum_{i=1}^{N} \left[\frac{2 m_i K(r_x, r_y, r_z)}{h_{x_i} h_{y_i} h_{z_i}} \right] , \tag{9.2}$$

where h_{x_i}, h_{y_i} and h_{z_i} are the kernel bandwidths which determine the degree of smoothing in each coordinate direction, $r_x = (X_i - x)/h_{x_i}$, $r_y = (Y_i - y)/h_{y_i}$, $r_z = Z_i/h_{z_i}$ with X_i, Y_i and Z_i being the position of particle i.

The horizontal kernel bandwidths for each particle are defined according to

$$h_{x_i} = h_{y_i} = 20000 \text{ m } + 0.8 \text{ ms}^{-1} \; t_t + 16500 \text{ m } \sqrt{t_t / 10800 \text{ s}} \, , \tag{9.3}$$

where t_t is the time (in seconds) since the release of particle i. This gives bandwidths of 20, 140 and 380 km after 0, 24 h and 96 h travel time, respectively. This function assumes a linear increase in horizontal trajectory position errors with travel time for long travel times and a faster increase at short travel times (Stohl and Seibert, 1998). Because this definition is based on rather small estimates of trajectory errors (Stohl, 1998), and because the estimated concentrations are most strongly affected by the particles closest to the receptor locations, results of a hypothetically perfect simulation are not spoiled by this definition of the bandwidths. But due to its smoothing effect, it can minimize the impact of small inaccuracies in the transport simulations.

The vertical kernel bandwidth for each particle is given by

$$h_{z_i} = 50 \text{ m } + 31.2 \text{ m } \sqrt{t_t/10800 \text{ s}} \, . \tag{9.4}$$

The maximum value for h_{z_i} is 150 m, coinciding with the minimum value of the PBL height.

The kernel is defined as

$$K(r_x, r_y, r_z) = \frac{15}{8\pi}(1 - r^2)I \, , \tag{9.5}$$

where $I = 1$ for $r^2 = r_x^2 + r_y^2 + r_z^2 < 1$ and $I = 0$ otherwise. The age spectra calculation is not done for this kernel output.

9.2.6 Concentration sampling

With both the uniform and the parabolic kernel, concentrations (or mixing ratios) C_{T_c} at time T_c are calculated as time-averages over the period $[T_c - \Delta T_c/2, T_C + \Delta T_c/2]$. ΔT_c must be specified by the user (variable `loutaver` read in from `COMMAND`). To calculate the time-averages, concentrations C_{T_s} at times T_s within $[T_c - \Delta T_c/2, T_C + \Delta T_c/2]$ are sampled at shorter intervals ΔT_s (variable `loutsample` read in from `COMMAND`) and are then divided by the number of samples taken:

$$C_{T_c} = \frac{1}{N} \sum_{i=1}^{N} C_{T_s} \, , \tag{9.6}$$

where $N = \frac{\Delta T_c}{\Delta T_s}$ is the number of samples that fits into time interval ΔT_c. ΔT_s must also be specified by the user. Both ΔT_c and ΔT_s must be multiples of the specified synchronisation interval (variable `lsynctime` read in from `COMMAND`). The shorter the sampling intervals ΔT_s, the more samples are taken to compute a single average concentration and the more accurate are thus the time-averaged concentrations.

9.3 Deposition fields

Wet and dry deposition fields are calculated on the same output grid as used for the concentrations or mixing ratios using the uniform kernel (subroutines `wetdepokernel` and `drydepokernel`). The deposited matter is accumulated over the course of a model run, i.e. it generally increases with model time. However, radioactive decay calculations are continued also for the deposited matter.

The uncertainty of the deposition fields is calculated in the same way as for the concentration fields, i.e., by calculating the deposition fields independently for `nclassunc` classes. Age spectra are optionally created, too. For deposition, the age is defined as the time between release and deposition, i.e. no ageing occurs at the ground.

Wet and dry deposition field output is written both to files `grid_conc_date` and `grid_pptv_date`.

9.4 Mass fluxes

Mass fluxes are calculated separately for eastward, westward, northward, southward, upward and downward motions. These gross fluxes include both the grid-scale motions as well as the contribution from subgrid-scale (i.e. turbulent diffusion, deep convection) winds. Since the flux calculation is done only every synchronization interval, it underestimates the gross fluxes somewhat, because the most rapid sub-grid scale motions may not be captured fully. However, net fluxes (e.g. difference between E- and W-flux) calculated from the gross fluxes in post-processing will be accurate.

Mass fluxes are determined for the centerlines of each output grid cell, e.g. vertical fluxes are calculated for motions across the half level of each output cell (in m agl). The mass fluxes caused by the individual particles are accumulated until the next output of the fields. Upon output, the fluxes are re-set to zero.

Chapter 10

Domain-filling option

10.1 General

If `mdomainfill` is set to 1, a domain-filling option is used. With this option, particles are not released at specific location(s). Instead, the longitudes and latitudes specified in the `RELEASES` files for the first release are used to set up a model domain (can be either global or limited). The total atmospheric mass in this domain is determined and the particles (number is taken from `RELEASES`) are then distributed in the model domain proportionally to the air density (subroutine `init_domainfill.f`). Subsequently, particles are allowed to move freely in the atmosphere. Note that all particles have the same mass.

If a limited domain is chosen, mass fluxes are determined in small grid boxes at the boundary of this domain (boundaries must be at least one grid box away from the boundaries of the meteorological input data). In the grid cells experiencing air flowing into the model domain, mass fluxes are accumulated over time and whenever the accumulated mass exceeds the mass of a particle, a new particle (or more, if required) is released at a randomly chosen position at the boundary of the box (soubroutine `boundcond_domainfill.f`). In contrary, particles are destroyed at the outflowing boundaries. Note that, due to the change of mass of the atmosphere in the model domain and due to the numerics, the number of particles used is not constant throughout the simulation. Therefore, `maxpart` should always be larger than the number of particles specified in the release box.

10.2 Stratospheric ozone tracer

If `mdomainfill` is set to 2, the domain-filling option is used to simulate a stratospheric ozone tracer. Particles are created as in the normal domain-filling option. The potential vorticity (PV) at a particle's position is then determined by interpolation from the ECMWF data. Particles initially located in the troposphere (PV<2 potential vorticity units (pvu)) are immediately dropped from the simulation (parameter `pvcrit` can be used to change this value). In contrast, stratospheric particles (PV>2 pvu) were given a mass according to:

$$M_{O_3} = M_{air}\, P\, C\, 48/29 \tag{10.1}$$

where M_{air} is the mass of air a particle represents, P is PV, $C = 60{\times}10^{-9}$ pvu^{-1} is the ratio between the ozone volume mixing ratio and PV in the stratosphere (see Stohl et al. [2000]). The factor 48/29 converts from volume mixing ratio to mass mixing ratio. Parameter `ozonescale` can be used to change the ozone/PV relationship. The particles are then allowed to advect through the stratosphere and into the troposphere according to the winds.

Chapter 11

References

Asman, W.A.H. (1995): Parameterization of below-cloud scavenging of highly soluble gases under convective conditions. *Atmos. Environ.*, 29, 1359-1368.

Baumann, K., and A. Stohl (1997): Validation of a long-range trajectory model using gas balloon tracks from the Gordon Bennett Cup 95. *J. Appl. Meteor.*, **36**, 711-720.

Businger, J.A., J.C. Wyngaard, Y. Izumi, and E.F. Bradley (1971): Flux-profile relationships in the atmospheric surface layer. *J. Atmos. Sci.*, **28**, 181-189.

De Baas, A.F., H. Van Dop, F.T.M. Nieuwstadt (1986): An application of the Langevin equation for inhomogeneous conditions to dispersion in a convective boundary layer. *Q. J. R. Meteorol. Soc.*, **112**, 165-180.

Beljaars, A.C.M., and A.A.M. Holtslag (1991): Flux parameterization over land surfaces for atmospheric models. *J. Appl. Meteor.*, **30**, 327-341.

Berkowicz, R., and L.P. Prahm (1982): Evaluation of the profile method for estimation of surface fluxes of momentum and heat. *Atmos. Environ.*, **16**, 2809-2819.

Businger, J.A., J.C. Wyngaard, Y. Izumi, and E.F. Bradley (1971): Flux-profile relationships in the atmospheric surface layer. *J. Atmos. Sci.*, **28**, 181-189.

Dorling, S.R., Davies, T.D. and Pierce, C.E. (1992) Cluster analysis: a technique for estimating the synoptic meteorological controls on air and precipitation chemistry - method and applications. *Atmos. Environ.***26A**, 2575-2581.

ECMWF (1995): *User Guide to ECMWF Products 2.1.* Meteorological Bulletin M3.2. Reading, UK.

Emanuel, K. A., 1991, A scheme for representing cumulus convection in large-scale models, *J. Atmos. Sci., 48*, 2313–2335.

Emanuel, K. A., and M. Živković-Rothman (1999): Development and evaluation of a convection scheme for use in climate models. *J. Atmos. Sci.*, **56**, 1766-1782.

Erisman, J.W., A. Van Pul, and P. Wyers (1994): Parametrization of surface resistance for the quantification of atmospheric deposition of acidifying pollutants and ozone. *Atmos. Environ.*, **28**, 2595-2607.

Forster, C., Wandinger, U., Wotawa, G., James, P., Mattis, I., Althausen, D., Simmonds, P., O'Doherty, S., Kleefeld, C., Jennings, S. G., Schneider, J., Trickl, T., Kreipl, S., Jäger, H., Stohl, A. (2001): Transport of boreal forest fire emissions from Canada to Europe. J. Geophys. Res. 106, 22,887-22,906.

Gupta, S., McNider, R.T., Trainer, M., Zamora, R.J., Knupp, K. and Singh, M.P. (1997): Nocturnal wind structure and plume growth rates due to inertial oscillations. *J. Appl. Meteor.*, **36**, 1050-1063.

Hanna, S.R. (1982): Applications in air pollution modeling. In: Nieuwstadt F.T.M. and H. van Dop (ed.): *Atmospheric Turbulence and Air Pollution Modelling*. D. Reidel Publishing Company, Dordrecht, Holland.

Hertel, O., J. Christensen, E.H. Runge, W.A.H. Asman, R. Berkowicz, M.F. Hovmand, and O. Hov (1995): Development and testing of a new variable scale air pollution model - ACDEP. *Atmos. Environ.*, **29**, 1267-1290.

Hubbe, J.M., Doran, J.C., Liljegren, J.C. and Shaw, W.J. (1997) Observations of spatial variations of boundary layer structure over the southern Great Plains cloud and radiation testbed. *J. Appl. Meteor.*, **36**, 1221-1231.

Kahl, J.D.W. (1996): On the prediction of trajectory model error. *Atmos. Environ.*, **30**, 2945-2957.

Legg, B.J., and M.R. Raupach (1982): Markov-chain simulation of particle dispersion in inhomogeneous flows: the mean drift velocity induced by a gradient in Eulerian velocity variance. *Boundary-Layer Meteorol.*, **24**, 3-13.

Legras, B., B. Joseph, and F. Lefevre (2003): Vertical diffusivity in the lower stratosphere from Lagrangian back-trajectory reconstructions of ozone profiles. *J. Geophys. Res.*, **108**, 4562, doi:10.1029/2002JD003045.

Maryon, R.H. (1998) Determining cross-wind variance for low frequency wind meander. *Atmos. Environ.*, **32**, 115-121.

McMahon, T.A., and P.J. Denison (1979): Empirical atmospheric deposition parameters - a survey. *Atmos. Environ.*, **13**, 571-585.

McNider, R.T., M.D. Moran, and R.A. Pielke (1988): Influence of diurnal and inertial boundary layer oscillations on long-range dispersion. *Atmos. Environ.*, **22**, 2445-2462.

Pruppacher, H.R., and J.D. Klett (1978): *Microphysics of clouds and precipitation.* D. Reidel Publishing Company, Dordrecht.

Rodean, H. (1996): Stochastic Lagrangian models of turbulent diffusion. *Meteorological Monographs*, **26** (48). American Meteorological Society, Boston, USA.

Ryall, D.B. and Maryon, R.H. (1997): Validation of the UK Met Office's NAME model against the ETEX dataset. In: Nodop, K. (editor): *ETEX Symposium on Long-Range Atmospheric Transport, Model Verification and Emergency Response*, European Commission EUR 17346, 151-154.

Seibert, P. (2001): Inverse modelling with a Lagrangian particle dispersion model: application to point releases over limited time intervals. In: Gryning, S. E., Schiermeier, F.A. (eds.): *Air Pollution Modeling and its Application XIV*. Proc. of ITM Boulder. New York: Plenum Press, 381-389.

Seibert, P. and A. Frank (2004): Source-receptor matrix calculation with a Lagrangian particle dispersion model in backward mode. *Atmos. Chem. Phys.*, **4**, 51-63. http://www.copernicus.org/EGU/acp/acp/4/51

Seibert, P., and A. Stohl, 2000, Inverse modelling of the ETEX-1 release with a Lagrangian particle model. In: Barone, G., Builtjes, P.J., Giunta, G. (Eds.): GLObal and REgional Atmospheric Modelling. Annali dell'Istituto Universitario Navale di Napoli, Special Issue, 95-105. Proc. 3rd GLOREAM Workshop, Sept.

Seibert, P., B. Krüger, and A. Frank, 2001, Parametrisation of convective mixing in a Lagrangian particle dispersion model, Proceedings of the 5th GLOREAM Workshop, Wengen, Switzerland, September 24 - 26.

Spichtinger, N., M. Wenig, P. James, T. Wagner, U. Platt, and A. Stohl (2001): Satellite detection of a continental-scale plume of nitrogen oxides from boreal forest fires, *Geophys. Res. Lett.28*, 4579-4582.

Slinn (1982): Predictions for particle deposition to vegetative canopies. *Atmos. Environ.*, **16**, 1785-1794.

Stohl, A. (1998) Computation, accuracy and applications of trajectories – a review and bibliography. *Atmos. Environ.*, **32**, 947-966.

Stohl, A., S. Eckhardt, C. Forster, P. James, N. Spichtinger, and P. Seibert (2002): A replacement for simple back trajectory calculations in the interpretation of atmospheric trace substance measurements. *Atmos. Environ.*, **36**, 4635-4648.

Stohl, A., C. Forster, S. Eckhardt, N. Spichtinger, H. Huntrieser, J. Heland, H. Schlager, S. Wilhelm, F. Arnold, and O. Cooper (2003): A backward modeling study of intercontinental pollution transport using aircraft measurements. *J. Geophys. Res., 108*, 4370, doi:10.1029/2002JD002862.

Stohl, A., Hittenberger, M., Wotawa, G. (1998): Validation of the Lagrangian particle dispersion model FLEXPART against large scale tracer experiment data. *Atmos. Environ.*, **24**, 4245-4264.

Stohl, A., and P. Seibert (1998): Accuracy of trajectories as determined from the conservation of meteorological tracers. *Q. J. R. Meteorol. Soc.*, **125**, 1465-1484.

Stohl, A., N. Spichtinger-Rakowsky, P. Bonasoni, H. Feldmann, M. Memmesheimer, H. E. Scheel, T. Trickl, S. H. Hbener, W. Ringer, and M. Mandl, The influence of stratospheric intrusions on alpine ozone concentrations. *Atmos. Environ.***34**, 1323-1354, 2000.

Stohl, A. and Thomson, D.J. (1999): A density correction for Lagrangian particle dispersion models. *Boundary-Layer Meteorol.*, **90**, 155-167.

Stohl, A., and T. Trickl (1999): A textbook example of long-range transport: Simultaneous observation of ozone maxima of stratospheric and North American origin in the free troposphere over Europe, *J. Geophys. Res.104*, 30,445-30,462.

Stohl, A., G. Wotawa, P. Seibert, and H. Kromp-Kolb (1995): Interpolation errors in wind fields as a function of spatial and temporal resolution and their impact on different types of kinematic trajectories. *J. Appl. Meteor.*, **34**, 2149-2165.

Stull, R.B. (1988): *An Introduction to Boundary Layer Meteorology.* Kluwer Academic Publishers, Dordrecht.

Telford, J. W., 1975, Turbulence, entrainment and mixing in cloud dynamics, *Pure Appl. Geophys., 113*, 1067 - 1084.

Thomson D.J. (1984): Random walk modelling of diffusion in inhomogeneous turbulence. *Q. J. R. Meteorol. Soc.*, **110**, 1107-1120.

Thomson D.J. (1987): Criteria for the selection of stochastic models of particle trajectories in turbulent flows. *J. Fluid Mech.*, **180**, 529-556.

Uliasz, M. (1994): Lagrangian particle dispersion modeling in mesoscale applications. In: Zannetti, P. (ed.): *Environmental Modeling, Vol. II.* Computational Mechanics Publications, Southampton, UK.

Velde van de, R.J., W.S. Faber, V.F. Katwijk van,, J.C.I. Kuylenstierna, H.J. Scholten., T.J.M. Thewessen, M. Verspuij, and M. Zevenbergen (1994): *The Preparation of a European Land Use Database.* National Institute of Public Health and Environmental Protection, Report nr 712401001, Bilthoven, The Netherlands.

Vogelezang, D.H.P., and A.A.M. Holtslag (1996): Evaluation and model impacts of alternative boundary-layer height formulations. *Boundary-Layer Meteorol.*, in press.

Wesely, M.L., and B.B. Hicks (1977): Some factors that affect the deposition rates of sulfur dioxide and similar gases on vegetation. *J. Air Poll. Contr. Assoc.*, **27**, 1110-1116.

Wesely, M.L. (1989): Parameterization of surface resistances to gaseous dry deposition in regional-scale numerical models. *Atmos. Environ.*, **23**, 1293-1304.

Wilson, D.J., and Flesch, T.K. (1993): Flow boundaries in random-flight dispersion models: enforcing the well-mixed condition. *J. Appl. Meteor.*, **32**, 1695-1707.

Wilson, J.D., B.J. Legg, and D.J. Thomson (1983): Calculation of particle trajectories in the presence of a gradient in turbulent-velocity scale. *Boundary-Layer Meteorol.*, **27**, 163-169.

Wilson, J.D., and Sawford, B.L. (1996): Review of Lagrangian stochastic models for trajectories in the turbulent atmosphere. *Boundary-Layer Meteorol.*, **78**, 191-210.

Wilson, J.D., G.W. Thurtell, and G.E. Kidd (1981): Numerical simulation of particle trajectories in inhomogeneous turbulence, II: Systems with variable turbulent velocity scale. *Boundary-Layer Meteorol.*, **21**, 423-441.

Wotawa, G., A. Stohl, and H. Kromp-Kolb (1996): Parameterization of the planetary boundary layer over Europe - a data comparison between the observation based OML preprocessor and ECMWF model data. *Contr. Atmos. Phys.*, **69**, 273-284.

Wotawa, G. and Stohl, A. (1997) Boundary layer heights and surface fluxes of momentum and heat derived from ECMWF data for use in pollutant dispersion models – problems with data accuracy. In: Gryning, S.-E., Beyrich, F. and E. Batchvarova (editors): The Determination of the Mixing Height – Current Progress and Problems. EURASAP Workshop Proceedings, 1-3 October 1997, Risø National Laboratory, Denmark.

Zannetti, P. (1992): Particle Modeling and Its Application for Simulating Air Pollution Phenomena. In: Melli P. and P. Zannetti (ed.): *Environmental Modelling.* Computational Mechanics Publications, Southampton, UK.

Appendix A

FLEXPART input files

A.1 The pathnames file

A file `pathnames` must exist in the working directory of FLEXPART . It contains the information, where input files are stored, and to which directory the output is directed:

```
/home/as/flexpart50/options/
/volc/as/contrace/modelresults/forward/
/volc/windcontrace/
/volc/windcontrace/AVAILABLE
/volc/nested/
/volc/nested/AVAILABLE
=========================================

Line 1: path where control files "COMMAND" and "RELEASES" are available
Line 2: name of directory where output files are generated
Line 3: path where meteorological fields are available (mother grid)
Line 4: full filename of "AVAILABLE"-file (mother grid)

Subsequent lines:
Line 2n+3: path where meteorological fields are available (nested grid n)
Line 2n+4: full filename of "AVAILABLE"-file (nested grid n)

Line below last pathname must be:
=========================================

The grids must be arranged such as that the coarse-scale nests
come before the fine-scale nests. Multiple nests of the same
nesting level are allowed. In that case, the order is arbitrary.
```

A.2 Files in directory windfields

The directory where the meteorological input data are stored, here called `windfields` (`/volc/windcontrace/` in the above example `pathnames` file), contains grib-code files containing the ECMWF data. All meteorological fields must have the same structure, i.e. the same computational domain and the same resolution. An example listing of this directory is given below.

```
AVAILABLE       EN01102806      EN01102815
EN01102800      EN01102809      EN01102818
EN01102803      EN01102812      EN01102821
```

The file names of the grib-code files and their validation dates and times (in UTC) must be listed in the file `AVAILABLE`. While it is practical to have this file reside in the same directory as the wind fields, this is no necessity and it can also be located elsewhere, as its file name is also given in the `pathnames` file.

```
DATE     TIME         FILENAME     SPECIFICATIONS
YYYYMMDD HHMISS

-------- ------      ----------      ----------
20011028 000000      EN01102800      ON DISC
20011028 030000      EN01102803      ON DISC
20011028 060000      EN01102806      ON DISC
20011028 090000      EN01102809      ON DISC
20011028 120000      EN01102812      ON DISC
20011028 150000      EN01102815      ON DISC
20011028 180000      EN01102818      ON DISC
20011028 210000      EN01102821      ON DISC
```

Nested wind fields must be stored in one or more different directory/ies, as specified in the `pathnames` file. For each nesting level, there must also be an AVAILABLE file and the different AVAILABLE files must all be consistent, i.e., contain entries for the same dates.

A.3 Files in directory options

The files in directory `options` are used to specify the model run. An example listing of `options` is given below.

```
AGECLASSES                  EMISSION_VARIATION_023.dat  landuse.asc   RELEASES.alternative
COMMAND                     EMISSION_VARIATION_025.dat  OUTGRID       RELEASES.reference
COMMAND.alternative         EMISSION_VARIATION_026.dat  OUTGRID_NEST  SPECIES
COMMAND.reference           EMISSION_VARIATION_027.dat  RECEPTORS     surfdata.t
EMISSION_VARIATION_008.dat  EMISSION_VARIATION_028.dat  RELEASES      surfdepo.t
```

The most important file is the `COMMAND` file which specifies (1) the simulation direction (either forward or backward), (2) the start and (3) the end time of the simulation, (4) the frequency T_c of the model output, (5) the averaging time ΔT_c of model output, and (6) the intervals ΔT_s at which concentrations are sampled, (7) the time constant for particle splitting Δt_s, (8) the synchronisation interval of FLEXPART , (9) the factor c_{tl} by which the time steps must be smaller than the Lagrangian time scale, and (10) the refinement factor for the time step used for solving the Langevin equation of the vertical component of the turbulent wind. If (9) (c_{tl}) is negative, the Langevin equations are solved with constant time steps according to the synchronisation interval. In that case, the value of (10) is arbitrary. The synchronisation interval is the minimum time interval used by the model for all activities (such as concentration calculations, wet deposition calculations, interpolation of data, mesoscale wind fluctuations or output of data) other than the simulation of turbulent transport and dry deposition (if ($c_{tl} > 0$). Further switches determine (11) whether concentrations, mixing ratios, residence times or plume trajectories (or combinations thereof) are to be calculated, (12) the option of particle position dump either at the end of or continuously during the simulation, (13) on/off of subgrid terrain effect parameterization, (14) on/off of deep convection parameterization, (15) on/off calculation of age spectra, (16) continuation of simulation from previous particle dump, (17) on/off for mass flux calculations and output, (18) on/off for the domain-filling option of FLEXPART , (19) an indicator that determines whether mixing ratios or concentrations at the receptor are to be modeled in a backward run, (20) on/off of additional compact dump of the positions of numbered particles, (21) on/off for the use of nested output fields.

Two versions of `COMMAND` may be used, which both can be read in by FLEXPART : the first contains formatted input (i.e., a mask to be filled for the various input options that must be filled in), the second contains largely unformatted input and is recommended for the more experienced FLEXPART user. The following example is for formatted input.

```
*******************************************************************************
*                                                                             *
*      Input file for the Lagrangian particle dispersion model FLEXPART       *
*                           Please select your options                        *
*                                                                             *
*******************************************************************************

1. __                3X, I2
    1
   DIRECTION         1 FOR FORWARD SIMULATION, -1 FOR BACKWARD SIMULATION

2. ________ ______   3X, I8, 1X, I6
   20040626 000000
   YYYYMMDD HHMISS   BEGINNING DATE OF SIMULATION

3. ________ ______   3X, I8, 1X, I6
   20040816 120000
   YYYYMMDD HHMISS   ENDING DATE OF SIMULATION

4. _____             3X, I5
    7200
   SSSSS             OUTPUT EVERY SSSSS SECONDS

5. _____             3X, I5
    7200
   SSSSS             TIME AVERAGE OF OUTPUT (IN SSSSS SECONDS)

6. _____             3X, I5
     900
   SSSSS             SAMPLING RATE OF OUTPUT (IN SSSSS SECONDS)

7. _________         3X, I9
   999999999
   SSSSSSSSS         TIME CONSTANT FOR PARTICLE SPLITTING (IN SECONDS)

8. _____             3X, I5
     900
   SSSSS             SYNCHRONISATION INTERVAL OF FLEXPART (IN SECONDS)

9.  ---.--           4X, F6.4
     -5.0
    CTL              FACTOR, BY WHICH TIME STEP MUST BE SMALLER THAN TL

10. ---              4X, I3
      4
    IFINE            DECREASE OF TIME STEP FOR VERTICAL MOTION BY FACTOR IFINE

11. -                4X, I1
    3
    IOUT             1 CONCENTRATION (RESIDENCE TIME FOR BACKWARD RUNS) OUTPUT, 2 MIXING RATIO OUTPUT, 3 BOTH,4 PLUME TRA

12. -                4X, I1
    2
    IPOUT            PARTICLE DUMP: 0 NO, 1 EVERY OUTPUT INTERVAL, 2 ONLY AT END
```

```
13. _                4X, I1
    1
    LSUBGRID         SUBGRID TERRAIN EFFECT PARAMETERIZATION: 1 YES, 0 NO

14. _                4X, I1
    1
    LCONVECTION      CONVECTION: 1 YES, 0 NO

15. _                4X, I1
    0
    LAGESPECTRA      AGE SPECTRA: 1 YES, 0 NO

16. _                4X, I1
    0
    IPIN             CONTINUE SIMULATION WITH DUMPED PARTICLE DATA: 1 YES, 0 NO

17. _                4X, I1
    0
    IFLUX            CALCULATE FLUXES: 1 YES, 0 NO

18. _                4X, I1
    2
    MDOMAINFILL      DOMAIN-FILLING TRAJECTORY OPTION: 1 YES, 0 NO, 2 STRAT. O3 TRACER

19. _                4X, I1
    1
    INDICATOR_BACKWARD 2=CONCENTRATION INPUT, OTHERWISE MIXING RATIO INPUT

20. _                4X, I1
    0
    MQUASILAG        QUASILAGRANGIAN MODE TO TRACK INDIVIDUAL PARTICLES: 1 YES, 0 NO

21. _                4X, I1
    1
    NESTED_OUTPUT    SHALL NESTED OUTPUT BE USED? YES, 0 NO

1. Simulation direction, 1 for forward, -1 for backward in time

2. Beginning date and time of simulation. Must be given in format
   YYYYMMDD HHMISS, where YYYY is YEAR, MM is MONTH, DD is DAY, HH is HOUR,
   MI is MINUTE and SS is SECOND. Current version utilizes UTC.

3. Ending date and time of simulation. Same format as 3.

4. Average concentrations are calculated every SSSSS seconds.

5. The average concentrations are time averages of SSSSS seconds
   duration. If SSSSS is 0, instantaneous concentrations are outputted.

6. The concentrations are sampled every SSSSS seconds to calculate the time
   average concentration. This period must be shorter than the averaging time.

7. Time constant for particle splitting. Particles are split into two
```

```
   after SSSSS seconds, 2xSSSSS seconds, 4xSSSSS seconds, and so on.

8. All processes are synchronized with this time interval (lsynctime).
   Therefore, all other time constants must be multiples of this value.
   Output interval and time average of output must be at least twice lsynctime.

9. CTL must be >1 for time steps shorter than the  Lagrangian time scale
   If CTL<0, a purely random walk simulation is done

10.IFINE=Reduction factor for time step used for vertical wind

11.IOUT determines how the output shall be made: concentration
   (ng/m3, Bq/m3), mixing ratio (pptv), or both, or plume trajectory mode,
   or concentration + plume trajectory mode.
   In plume trajectory mode, output is in the form of average trajectories.

12.IPOUT determines whether particle positions are outputted (in addition
   to the gridded concentrations or mixing ratios) or not.
   0=no output, 1 output every output interval, 2 only at end of the
   simulation

13.Switch on/off subgridscale terrain parameterization (increase of
   mixing heights due to subgridscale orographic variations)

14.Switch on/off the convection parameterization

15.Switch on/off the calculation of age spectra: if yes, the file AGECLASSES
   must be available

16. If IPIN=1, a file "partposit_end" from a previous run must be available in
    the output directory. Particle positions are read in and previous simulation
    is continued. If IPIN=0, no particles from a previous run are used

17. If IFLUX is set to 1, fluxes of each species through each of the output
    boxes are calculated. Six fluxes, corresponding to northward, southward,
    eastward, westward, upward and downward are calculated for each grid cell of
    the output grid. The control surfaces are placed in the middle of each
    output grid cell. If IFLUX is set to 0, no fluxes are determined.

18. If MDOMAINFILL is set to 1, the first box specified in file RELEASES is used
    as the domain where domain-filling trajectory calculations are to be done.
    Particles are initialized uniformly distributed (according to the air mass
    distribution) in that domain at the beginning of the simulation, and are
    created at the boundaries throughout the simulation period.

19. INDICATOR_BACKWARD indicates how the "mass" input in RELEASES shall be
    interpreted in backward runs. If INDICATOR_BACKWARD is set to 2, "mass"
    is interpreted as concentration. Otherwise, it is interpreted as mixing
    ratio.

20. MQUASILAG indicates whether particles shall be numbered consecutively (1) or
    with their release location number (0). The first option allows tracking of
    individual particles using the partposit output files
```

```
21. NESTED_OUTPUT decides whether model output shall be made also for a nested
    output field (normally with higher resolution)
```

The file `OUTGRID` specifies the output grid. The maximum allowed number of output levels is set by parameter `maxzgrid` in file `includepar`. The maximum dimensions in x and y by parameters `maxxgrid` and `maxygrid`.

```
*******************************************************************************
*                                                                             *
*      Input file for the Lagrangian particle dispersion model FLEXPART       *
*                       Please specify your output grid                       *
*                                                                             *
*******************************************************************************

1.  ------.----       4X,F11.4
       -10.0000       GEOGRAFICAL LONGITUDE OF LOWER LEFT CORNER OF OUTPUT GRID
    OUTLONLEFT        (left boundary of the first grid cell - not its centre)


2.  ------.----       4X,F11.4
        40.0000       GEOGRAFICAL LATITUDE OF LOWER LEFT CORNER OF OUTPUT GRID
    OUTLATLOWER       (lower boundary of the first grid cell - not its centre)

3.  -----             4X,I5
      101             NUMBER OF GRID POINTS IN X DIRECTION (= No. of cells + 1)
    NUMXGRID

4.  -----             4X,I5
       47             NUMBER OF GRID POINTS IN Y DIRECTION (= No. of cells + 1)
    NUMYGRID

5.  ------.---        4X,F10.3
         0.500        GRID DISTANCE IN X DIRECTION
    DXOUTLON

6.  ------.---        4X,F10.3
         0.500        GRID DISTANCE IN Y DIRECTION
    DYOUTLAT

7.  -----.-           4X, F7.1
      100.0
    LEVEL 1           HEIGHT OF LEVEL (UPPER BOUNDARY)

8.  -----.-           4X, F7.1
      300.0
    LEVEL 2           HEIGHT OF LEVEL (UPPER BOUNDARY)

9.  -----.-           4X, F7.1
      600.0
```

```
    LEVEL 3           HEIGHT OF LEVEL (UPPER BOUNDARY)

10. -----.-           4X, F7.1
     1000.0
    LEVEL 4           HEIGHT OF LEVEL (UPPER BOUNDARY)

11. -----.-           4X, F7.1
     2000.0
    LEVEL 5           HEIGHT OF LEVEL (UPPER BOUNDARY)

12. -----.-           4X, F7.1
     3000.0
    LEVEL 6           HEIGHT OF LEVEL (UPPER BOUNDARY)
```

In order to define the grid for a nested output field, the file `OUTGRID_NEST` must exist. It has the same format as file `OUTGRID`, but does not contain the vertical level information:

```
*******************************************************************************
*                                                                             *
*      Input file for the Lagrangian particle dispersion model FLEXPART       *
*                       Please specify your output grid                       *
*                                                                             *
*******************************************************************************

1.  ------.----       4X,F11.4
      -125.0000       GEOGRAFICAL LONGITUDE OF LOWER LEFT CORNER OF OUTPUT GRID
    OUTLONLEFT        (left boundary of the first grid cell - not its centre)

2.  ------.----       4X,F11.4
        25.0000       GEOGRAFICAL LATITUDE OF LOWER LEFT CORNER OF OUTPUT GRID
    OUTLATLOWER       (lower boundary of the first grid cell - not its centre)

3.  -----             4X,I5
        1             NUMBER OF GRID POINTS IN X DIRECTION (= No. of cells + 1)
    NUMXGRID

4.  -----             4X,I5
        1             NUMBER OF GRID POINTS IN Y DIRECTION (= No. of cells + 1)
    NUMYGRID

5.  ------.-----      4X,F12.5
         0.33333      GRID DISTANCE IN X DIRECTION
    DXOUTLON

6.  ------.-----      4X,F12.5
         0.25000      GRID DISTANCE IN Y DIRECTION
    DYOUTLAT
```

RECEPTORS specifies the receptor locations for which the parabolic kernel method shall be applied to calculate air concentrations. The maximum number of receptor sites is set by parameter maxreceptor in file includepar.

```
*******************************************************************************
*                                                                             *
*      Input file for the Lagrangian particle dispersion model FLEXPART       *
*                       Please specify your receptor points                   *
*     For the receptor points, ground level concentrations are calculated     *
*                                                                             *
*******************************************************************************
1.  ----------------  4X,A16
    F15               NAME OF RECEPTOR POINT
    RECEPTORNAME

2.  ------.----       4X,F11.4
         6.1333       GEOGRAFICAL LONGITUDE
    XRECEPTOR

3.  ------.----       4X,F11.4
        49.0833       GEOGRAFICAL LATITUDE
    YRECEPTOR
===============================================================================
1.  ----------------  4X,A16
    NL01              NAME OF RECEPTOR POINT
    RECEPTORNAME

2.  ------.----       4X,F11.4
         5.7833       GEOGRAFICAL LONGITUDE
    XRECEPTOR

3.  ------.----       4X,F11.4
        50.9167       GEOGRAFICAL LATITUDE
    YRECEPTOR
===============================================================================
```

RELEASES defines the release specifications. In the first input line, the number N of emitted species is defined (1 in the example below). At all locations, the same species must be released. The next N input lines give a cross-reference to file SPECIES, where the physical and chemical properties of the released species are given (also the temporal variations of emissions is defined for each species). Then follows a list of release sites (maximum specified by parameter maxpoint in file includepar), for each of which the release characteristics must be entered: the beginning and the ending time of the release, geographical coordinates of the lower left and upper right corners of the release location, type of vertical coordinate (above ground level, or above sea level), lower level and upper level of source box, the number of particles to be used, and the total mass emitted. Note that the mass entry must be repeated N times, one mass per species released. Finally, a name is assigned to each release point.

The particles are released from random locations within a four-dimensional box extending from the lower to the upper level above a rectangle (on a lat/lon grid) defined by the geographical coordinates, and between the release's start and end. With some identical coordinates, line or point sources can be specified, too.

As for COMMAND, the RELEASES file can be provided formatted or unformatted. The example below shows the formatted version.

```
*****************************************************************************
*                                                                           *
*                                                                           *
*                                                                           *
*   Input file for the Lagrangian particle dispersion model FLEXPART        *
*                        Please select your options                         *
*                                                                           *
*                                                                           *
*                                                                           *
*****************************************************************************
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  1
___                          i3    Total number of species emitted

 24
___                          i3    Index of species in file SPECIES

=============================================================================
20011028  150007
________ ______              i8,1x,i6 Beginning date and time of release

20011028  150046
________ ______              i8,1x,i6 Ending date and time of release

   9.4048
____.____                    f9.4  Longitude [DEG] of lower left corner

  48.5060
____.____                    f9.4  Latitude [DEG] of lower left corner

   9.5067
____.____                    f9.4  Longitude [DEG] of upper right corner

  48.5158
____.____                    f9.4  Latitude [DEG] of upper right corner

        2
_________                    i9    1 for m above ground, 2 for m above sea level, 3 for pressure in hPa

 6933.60
_____.___                    f10.3 Lower z-level (in m agl or m asl)

 6950.40
_____.___                    f10.3 Upper z-level (in m agl or m asl)

    20000
_________                    i9    Total number of particles to be released

1.0000E00
_.____E__                    e9.4  Total mass emitted

FLIGHT_11242
________________________________________   character*40 comment
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

```
20011028  150047
________ ______           i8,1x,i6 Beginning date and time of release

20011028  150107
________ ______           i8,1x,i6 Ending date and time of release

   9.3038
____.____                 f9.4  Longitude [DEG] of lower left corner

  48.5158
____.____                 f9.4  Latitude [DEG] of lower left corner

   9.4048
____.____                 f9.4  Longitude [DEG] of upper right corner

  48.5906
____.____                 f9.4  Latitude [DEG] of upper right corner

        2
_________                 i9    1 for m above ground, 2 for m above sea level, 3 for pressure in hPa

 6833.50
_____.___                 f10.3 Lower z-level (in m agl or m asl)

 6950.40
_____.___                 f10.3 Upper z-level (in m agl or m asl)

    20000
_________                 i9    Total number of particles to be released

1.0000E00
_.____E__                 e9.4  Total mass emitted

FLIGHT_11185
________________________________________   character*40 comment
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Since $\mathrm{F_L e X^p_a R_T}$ version 6.0, emission factors can be defined that change the temporal variation of particle releases. This is useful, for instance, to simulate the typical daily and weekly cycle of anthropogenic emissions. The emission factors are given in files `EMISSION_VARIATION_nnn.dat`, where `nnn` is the species number defined in file `RELEASES`. If file `EMISSION_VARIATION_nnn.dat` does not exist, emission rates for species `nnn` are taken as constant. Release rates can vary with the hour of the day and with the day of the week, according to the local time at the release location. Emission factors must be 1 on average. 24 hourly as well as 7 daily values must be specified. Furthermore, different disaggregation factors must be given for area sources and for point sources. $\mathrm{F_L e X^p_a R_T}$

distinguishes between the two using the lower altitude of the release box: area sources are assumed to start below 0.5 m above the ground, whereas point sources are assumed to be higher. Please note that when this option is used, it is not so easy to determine the maximum number of particles present at a particular time of the model run. It might then be necessary to increase the parameter `maxpart` to a higher value than what would otherwise be needed. The following is an example for an `EMISSION_VARIATION_nnn.dat` file.

```
hr_start nox_area  nox_point
 0       0.578       0.845      0-1 local time
 1       0.491       0.806      1-2 local time
 2       0.428       0.786
 3       0.329       0.779
 4       0.384       0.793
 5       0.485       0.832
 6       0.763       0.895
 7       1.103       0.977
 8       1.084       1.031
 9       1.047       1.071
10       1.096       1.105
11       1.196       1.118
12       1.298       1.131
13       1.357       1.136
14       1.447       1.143
15       1.565       1.141
16       1.636       1.133
17       1.662       1.118
18       1.401       1.097
19       1.168       1.091
20       1.031       1.079
21       0.926       1.036
22       0.816       0.966
23       0.709       0.892      23-24 local time
week_day nox_area  nox_point
1        1.060       1.000      Monday
2        1.060       1.000      Tuesday
3        1.060       1.000      Wednesday
4        1.060       1.000      Thursday
5        1.060       1.000      Friday
6        0.900       1.000      Saturday
7        0.800       1.000      Sunday
```

AGECLASSES provides the times for the age class calculation. In the first data line, the number n of age classes is set, and ages are listed in the following n lines. The entries specify the end times (in seconds) of the respective intervals to be used, the first one starting at zero seconds. Particles are dropped from the simulation once they exceed the maximum age. Even if no age classes are needed, this option (with the number of age classes set to 1) can be useful to determine the age at which particles are removed from the simulation.

```
*********************************************
*                                           *
*Lagrangian particle dispersion model FLEXPART *
*         Please select your options        *
*                                           *
*This file determines the ageclasses to be used*
*                                           *
*Ages are given in seconds. The first class    *
*starts at age zero and goes up to the first   *
*age specified. The last age gives the maximum *
*time a particle is carried in the simulation. *
*                                           *
*********************************************
 6          Integer        Number of age classes
43200       Integer        Age class 1
86400       Integer        Age class 2
129600
172800
259200
345600
```

SPECIES is a list of species and their physico-chemical properties from which the user can select some for the simulation. Entries are the half life (due to radioactive or chemical decay), wet deposition information (A and B are the factors defined by equation 7.3), dry deposition information for gases ($D = D_{H_2O}/D_i$, D_{H_2O} is the diffusivity of water vapor and D_i is the diffusivity of the species, H is the effective Henry's constant, and $f0$ varies between 0 and 1 and gives the reactivity of a species relative to that of ozone. For nonreactive species $f0$ is 0, for slightly reactive it is 0.1 and for highly reactive it is 1.), dry depsosition information for particulates (rho specifies the density of the substance, $dquer$ its mean diameter $\overline{d_p}$, and $dsig$ the measure of variation σ_p). Radioactive decay is switched off by specifying a negative half life, wet deposition is switched off by specifying negative A, dry deposition of gases is switched off by negative D, dry deposition of particles is switched off by negative rho. If no detailed information for deposition velocity

calculation is available, a constant deposition velocity *vd* (cm s^{-1}) can be used. Finally, *molweight* gives the molecular weight of the species, which is needed for mixing ratio output.

```
*****************************************************************************
*                                                                           *
*     Input file for the Lagrangian particle dispersion model FLEXPART      *
*               Definition file of chemical species/radionuclides           *
*                                                                           *
*****************************************************************************
```

		Radioactivity	Wet depo		Dry depo (gases)			Dry depo (particles)			Dry depo	
	SPECIES	HALF LIFE [s]	A	B	D	H	f0	rho	dquer	dsig	vd	molweight
1	TRACER	-999.9	-9.9E-09		-9.9			-9.9E09			-9.99	350.00
2	O3	-999.9	-9.9E-09		1.5	1.0e-02	1.0	-9.9E09			-9.99	48.00
3	NO	-999.9	8.0E-06	0.62	1.2	2.0e-03	0.0	-9.9E09			-9.99	30.00
4	NO2	-999.9	1.0E-05	0.62	1.6	1.0e-02	0.1	-9.9E09			-9.99	46.00
5	HNO3	-999.9	8.0E-04	0.62	1.9	1.0e+14	0.0	-9.9E09			-9.99	63.00
6	HNO2	-999.9	-9.9E-09		1.6	1.0e+05	0.1	-9.9E09			-9.99	47.00
7	H2O2	-999.9	1.0E-04	0.62	1.4	1.0e+05	1.0	-9.9E09			-9.99	34.00
8	SO2	-999.9	-9.9E-09	0.62	2.0	1.0e+05	0.0	-9.9E09			-9.99	64.00
9	HCHO	-999.9	-9.9E-09		1.3	6.0e+03	0.0	-9.9E09			-9.99	30.00
10	PAN	-999.9	-9.9E-09		2.6	3.6e+00	0.1	-9.9E09			-9.99	121.00
11	NH3	-999.9	-9.9E-09		1.1	2.0e+14	0.0	-9.9E09			-9.99	17.00
12	SO4-aero	-999.9	1.0E-04	0.80	-9.9			2.0E03	4.0E-7	3.0E-1	-9.99	-9.99
13	NO3-aero	-999.9	1.0E-04	0.80	-9.9			2.0E03	4.0E-7	3.0E-1	-9.99	-9.99
14	I2-131	691200.0	8.0E-05	0.62	2.7	1.0e+05	0.1	-9.9E09			-9.99	-9.99
15	I-131	691200.0	1.0E-04	0.80	-9.9			2.5E03	6.0E-7	3.0E-1	-9.99	-9.99
16	Cs-137	-999.9	1.0E-04	0.80	-9.9			2.5E03	6.0E-7	3.0E-1	-9.99	-9.99
17	Y-91	5037120.0	1.0E-04	0.80	-9.9			2.5E03	6.0E-7	3.0E-1	-9.99	-9.99
18	Ru-106	31536000.0	1.0E-04	0.80	-9.9			2.5E03	6.0E-7	3.0E-1	-9.99	-9.99
19	Kr-85	-999.9	-9.9E-09		-9.9			-9.9E09			-9.99	-9.99
20	Sr-90	-999.9	1.0E-04	0.80	-9.9			2.5E03	6.0E-7	3.0E-1	-9.99	-9.99
21	Xe-133	198720.0	-9.9E-09		-9.9			-9.9E09			-9.99	-9.99
22	CO	-999.9	-9.9E-09		-9.9			-9.9E09			-9.99	28.00
23	NO2TRACER	-999.9	-9.9E-09		-9.9			-9.9E09			-9.99	46.00
24	AIRTRACER	-999.9	-9.9E-09		-9.9			-9.9E09			-9.99	29.00

`landuse.asc` contains the landuse inventory, and `surfdata.t`, shown below, gives the roughness lengths for each landuse class:

```
8 landuse categories are related with Leaf Area Index and roughness length
-----------------------------------------------------------
landuse    LAI         z0(m)       Albedo     comment
-----------------------------------------------------------
 1         0.50        0.10         0.18      Grassland for agricultural use
 2         2.00        0.15         0.10      Arable land
 3         3.00        0.30         0.18      Permanent crops
 4         7.00        0.60         0.15      Forest
 5         0.00        0.10         0.12      Inland water
 6         0.20        0.70         0.20      Urban areas
 7         1.00        0.10         0.15      Other
 8         0.00        0.10         0.12      Ocean
```

`surfdepo.t` gives the resistances needed for the parameterization of dry deposition of gases for the eight landuse classes and five seasonal categories. This file must not be changed by the user.

```
================================================================================
INPUT RESISTANCES (s/m) FOR THE COMPUTATION OF SURFACE RESISTANCES TO
DRY DEPOSITION
================================================================================
AFTER WESELY, 1989
================================================================================
1 to 8: Landuse types
================================================================================
Values are tabulated for 5 seasonal categories:
1     Midsummer with lush vegetation
2     Autumn with unharvested cropland
3     Late autumn after frost, no snow
4     Winter, snow on ground and subfreezing
5     Transitional spring with partially green short annuals
================================================================================
              1       2       3       4       5       6       7       8
--------------------------------------------------------------------------------
ri          120.     60.     65.     90.   9999.   9999.    150.   9999.     1
rlu        2000.   2000.   2000.   2000.   9999.   9999.   4000.   9999.
rac         100.    200.    365.   2000.      0.    100.    200.      0.
rgss        350.    150.    230.    500.      0.    400.    400.      0.
rgso        200.    150.    170.    200.   2000.    300.    200.   2000.
rcls       2000.   2000.   2000.   2000.   9999.   9999.   4000.   9999.
rclo       1000.   1000.   1000.   1000.   9999.   9999.   1000.   9999.
--------------------------------------------------------------------------------
```

```
ri         9999.   9999.   9999.    500.   9999.   9999.   9999.   9999.     2
rlu        9000.   9000.   9000.   5500.   9999.   9999.   9000.   9999.
rac         100.    150.    270.   1710.      0.    100.    140.      0.
rgss        350.    200.    285.    500.      0.    400.    400.      0.
rgso        200.    150.    170.    200.   2000.    300.    200.   2000.
rcls       9000.   9000.   9000.   3270.   9999.   9999.   9000.   9999.
rclo        400.    400.    400.    570.   9999.   9999.    400.   9999.
------------------------------------------------------------------------------
ri         9999.   9999.   9999.    500.   9999.   9999.   9999.   9999.     3
rlu        9000.   9999.   9470.   5500.   9999.   9999.   9000.   9999.
rac         100.     10.     20.   1330.      0.    100.    120.      0.
rgss        350.    150.    230.    500.      0.    400.    400.      0.
rgso        200.    150.    170.    200.   2000.    300.    200.   2000.
rcls       9000.   9999.   9470.   4500.   9999.   9999.   9000.   9999.
rclo        400.   1000.    570.    570.   9999.   9999.    600.   9999.
------------------------------------------------------------------------------
ri         9999.   9999.   9999.    800.   9999.   9999.   9999.   9999.     4
rlu        9999.   9999.   9999.   1200.   9999.   9999.   9000.   9999.
rac          10.     10.     20.   1330.      0.    100.     50.      0.
rgss        100.    100.    100.    100.      0.    100.     50.      0.
rgso       3500.   3500.   3500.   3500.   2000.    600.   3500.   2000.
rcls       9999.   9999.   9470.    390.   9999.   9999.   9000.   9999.
rclo       1000.   1000.    570.    632.   9999.   9999.    800.   9999.
------------------------------------------------------------------------------
ri          240.    120.    130.    180.   9999.   9999.    300.   9999.     5
rlu        4000.   4000.   4000.   2670.   9999.   9999.   8000.   9999.
rac          80.     50.    100.   1500.      0.    100.    120.      0.
rgss        350.    150.    230.    500.      0.    500.    400.      0.
rgso        200.    150.    170.    200.   2000.    300.    200.   2000.
rcls       4000.   4000.   4000.   2670.   9999.   9999.   8000.   9999.
rclo        500.   1000.    670.    750.   9999.   9999.    800.   9999.
------------------------------------------------------------------------------
```

A.4 Output files

The output files, which have been described in section 9.1, are all written to one directory, which is specified in the `pathnames` file.

Appendix B

Source code description

Each subroutine and function used by the model is commented. Here, only a short listing and description of the subroutines and a flow chart are presented.

```
SHORT DESCRIPTION OF THE SOURCE CODE FILES USED BY THE FLEXPART MODEL
---------------------------------------------------------------------
FLEXPART.f        main program, manages reading of run specifications, etc.;
                  last call is to timemanager.f, which manages the actual model
                  integration
advance.f         advection of the particles by grid-scale and turbulent winds;
                  solves the Langevin equations
assignland.f      assigns the fractions of 8 landuse classes to each ECMWF grid
                  point
boundcond_domainfill.f  creates particles at the inflowing/destroys particles
                  at the outflowing boundary, if domain-filling option is used
calcfluxes.f      calculates the gross N-S, E-W and up-down mass fluxes
                  resulting from the motion of a single particle
calcmatrix.f      interface to the convection scheme (convect.f) by K. Emanuel
calcmatrix_nests.f same as calxmatrix.f, but for the nested coordinates
calcpar.f         calculates boundary layer parameters: friction velocity,
                  convective velocity scale, Obukhov length, PBL height, partly
                  by calling other routines
calcpar_nests.f   same as calcpar.f, but for the nested grids
calcpv.f          calculates potential vorticity for mother domain grid points
calcpv_nests.f    same as calcpv.f, but for the nested grids
caldate.f         computes calendar date from Julian date
centerofmass.f    calculates the center of mass of n points on the Earth
clustering.f      cluster analysis of particle position
cmapf1.0.f        Program package for coordinate transformations between
                  latitude/longitude grid and polar stereographic projection
conccalc.f        calculates concentrations or residence times and/or
                  mixing ratios on the output grid and at receptor locations
concoutput.f      writes out concentrationsor residence times and/or
                  mixing ratios on the output grid and at receptor locations
                  (files grid_conc_date, grid_pptv_date, grid_time_date)
concoutput_nest.f same as concoutput.f, but for the nested output grid
convect.f         convection scheme written by Kerry Emanuel (slightly modified)
```

```
convmix.f          handles all the calculations related to convective mixing
coordtrafo.f       transformation of release point coordinates from geografical
                   to grid coordinates
distance.f         calculates distance (km) between 2 points on Earth's surface
                   given their positions in degrees
distance2.f        calculates distance (km) between 2 points on Earth's surface
                   given their positions in radians
drydepokernel.f    calculates gridded dry deposition field using a uniform kernel
drydepokernel_nest.f same as drydepokernel.f, but for the nested output grid
erf.f              error function
ew.f               calculation of saturation water vapor pressure for given
                   air temperature
fluxoutput.f       writes out mass fluxes on the output grid (grid_flux_date)
getfields.f        manages reading of new wind fields if required, and calls all
                   routines that do calculations on the input grids
getrb.f            computes quasilaminar sublayer resistance to dry deposition
                   of gases
getrc.f            calculates surface resistance to dry deposition of gases
getvdep.f          calculates dry deposition velocities
gridcheck.f        determines grid domain, resolution, etc., from GRIB file and
                   checks whether dimensions comply with settings in includepar
gridcheck_nests.f  same as gridcheck.f, but for the nested grids
hanna.f            parameterization of the turbulent velocity variance and
                   the respective timescales; Langevin equation for w'/sigw
hanna1.f           alternative model to hanna.f: Langevin equation for w'
                   drift correction velocity after Wilson
hanna_short.f      same as hanna.f, but only for the vertical wind
includecom         include file containing important global variables
includeconv        include file containing variables used in the convection scheme
includeinterpol    include file containing variables used for interpolation in
                   routines called from advance.f
includehanna       include file used for turbulence parameterization
includepar         include file containing all globally used parameters
init_domainfill.f  distributes particles homogeneously in the computation domain
initialize.f       initializes the turbulence statistics for a particle
interpol_all.f     interpolates all data needed in advance.f for two levels
                   within the PBL and for all surface data
interpol_all_nests.f  same as interpol_all.f, but for the nested grids
interpol_misslev.f interpolates for a level in the PBL that was not provided
                   by interpol_all.f, because particle has moved out of the
                   two sandwiched layers
interpol_misslev_nests.f same as interpol_misslev.f, but for the nested grids
interpol_rain.f    interpolates large-scale and convective precipitation,
                   and of total cloud cover
interpol_rain_nests.f same as interpol_rain.f, but for the nested grids
interpol_vdep.f    interpolates the deposition velocity
interpol_vdep_nests.f  same as interpol_vdep.f, but for the nested grids
interpol_wind.f    interpolates winds above the PBL and calculates the wind's
                   standard deviation for grid points around particle location
interpol_wind_nests.f   same as interpol_wind.f, but for the nested grids
interpol_wind_short.f  interpolates winds above the PBL
interpol_wind_short_nests.f  same as interpol_wind_short.f, but for nested grids
ipsort.f           sorts particles according to their coordinates
juldate.f          compute julian date from calendar date
```

```
mean.f              calculates the mean and standard deviation of a field
obukhov.f           calculates Obukhov length from surface heat flux
openouttraj.f       opens the output files for the plume trajectory output
openreceptors.f     opens output files for receptor location output, and writes
                    out some basic information (location and names of receptors)
                    (files receptor_conc and receptor_pptv)
outgrid_init.f      initializes the output grid and calculates volume and area
                    of output grid cells
outgrid_init_nest.f same as outgrid_init.f, but for the nested output grid
part0.f             calculates time-independent factors for the dry deposition
                    of particles
partdep.f           calculates dry deposition velocities for particles
partoutput.f        writes out the particle positions to files partposit_date, or
                    partposit_end
partoutput_short.f  dumps particle output in a compact format; particles are
                    uniquely numbered
pbl_profile.f       calculates surface stress and sensible heat flux using
                    the profile method
plumetraj.f         calculates a plume centroid trajectory and manages particle
                    clustering; writes output to file trajectories.txt
psih.f              stability correction term for heat
psim.f              stability correction term for momentum
qvsat.f             saturation specif. humidity at given pressure and temperature
raerod.f            calculation of the aerodynamic resistance ra to dry deposition
random.f            subroutines needed for generation of normally distriubted
                    random numbers
readageclasses.f    reads number and time intervals of age classes to be used
readavailable.f     reads, which wind fields are available during the modelling
                    period
readcommand.f       reads basic user commands, e.g. start and end of simulation,
                    frequency of output, etc.
readdepo.f          reads parameters needed for dry deposition of gases
readlanduse.f       reads landuse inventory and the respective roughness lengths
readoutgrid.f       reads the definition of the output domain
readoutgrid_nest.f  same as readoutgrid.f, but for the nested output grid
readpartpositions.f reads particle positions from a previous run's
                    partposit_end file, in order to initialize particle positions
                    for the new run
readpaths.f         reads the path names of input/output files and wind fields
readreceptors.f     reads names and coordinates of receptor points for which the
                    parabolic kernel shall be applied
readreleases.f      reads release specifications (start and end of release,
                    coordinates, number of particles, masses released, etc.)
readspecies.f       reads physical and chemical properties of gases and aerosols
readwind.f          reads the ECMWF meteorological data fields; contains the
                    calls to GRIB decoding routines
readwind_nests.f    same as readwind.f, but for the nested grids
redist.f            redistributes particles according to convective fluxes
redist_nests.f      same as redist.f, but for the nested grids
releaseparticles.f  releases the particles during the simulation; called every
                    lsynctime seconds
richardson.f        calculates convective velocity scale and PBL height using
                    critical Richardson number concept
scalev.f            computes friction velocity from surface stress
```

```
shift_field.f      shifts global fields by nxshift grid cells, and repeats
                   westernmost grid points at easternmost domain boundary
shift_field_0.f    same as shift_field.f, but for some fields with 2 dimensions
                   only (e.g. topography)
sigma.f            calculates the standard deviation of a 1-D field
skplin.f           skips n lines of a given input file
timemanager.f      time management of FLEXPART model simulation: determines when
                   particles are released and advected, when concentrations
                   calculations are done and concentrations written out, and
                   calls all the respective subroutines
verttransform.f    transforms 3-d fields from eta coordinates to Cartesian
                   terrain-following coordinates
verttransform_nests.f  same as verttransform.f, but for the nested grids
wetdepo.f          calculates the wet deposition
wetdepokernel.f    assigns the deposition from an individual particle to the
                   deposition fields using a uniform kernel of bandwidths dx, dy
wetdepokernel_nest.f same as wetdepokernel.f, but for the nested output grid
windalign.f        transforms turbulent velocities from along- and cross-wind
                   components to Cartesian u and v components
writeheader.f      creates the file header and writes important information about
                   the model run in it
writeheader_nest.f same as writeheader.f, but for the nested output grid
```

```
*****************************************************************************
*                   FLEXPART model calling structure                        *
*****************************************************************************

FLEXPART --> gasdev1 --> ran3
         --> readpaths
         --> readcommand --> juldate
                         --> skplin
         --> readageclasses
         --> readavailable --> juldate
         --> gridcheck
         --> gridcheck_nests
         --> readoutgrid --> skplin
         --> readoutgrid_nest --> skplin
         --> readreceptors
         --> readspecies
         --> readlanduse
         --> assignland
         --> readreleases --> skplin
                          --> part0 --> erf
         --> readdepo
         --> coordtrafo
         --> readpartpositions
         --> writeheader
         --> writeheader_nest
         --> openreceptors
         --> openouttraj
         --> outgrid_ini
         --> outgrid_ini_nest
         --> timemanager --> wetdepo --> interpol_rain
                                     --> interpol_rain_nests
                                     --> wetdepokernel
                                     --> wetdepokernel_nest
                         --> convmix --> ipsort
                                     --> calcmatrix       --> convect --> tlift
                                     --> calcmatrix_nests --> convect --> tlift
                                     --> redist
                                     --> redist_nests
                                     --> calcfluxes
                         --> getfields --> see %1
                         --> init_domainfill
                         --> boundcond_domainfill
                         --> releaseparticles --> caldate
                                              --> ran1 (random.f)
                         --> convmix --> ipsort
                                     --> calcmatrix       --> convect --> tlift
                                     --> calcmatrix_nests --> convect --> tlift
                                     --> redist
                                     --> redist_nests
                                     --> calcfluxes
                         --> conccalc
                         --> partoutput_short
                         --> concoutput --> caldate
                                        --> mean
```

```
                        --> concoutput_nest --> caldate
                                            --> mean
                        --> plumetraj --> clustering --> distance2
                                      --> centerofmass
                                      --> mean
                                      --> distance
                        --> fluxoutput --> caldate
                        --> partoutput --> caldate
                        --> conccalc
                        --> initialize --> same calls as advance
                        --> advance --> ran3
                                    --> interpol_all --> sigma
                                    --> interpol_all_nests --> sigma
                                    --> interpol_misslev --> sigma
                                    --> interpol_misslev_nests --> sigma
                                    --> hanna or hanna1
                                    --> hanna_short
                                    --> interpol_vdep
                                    --> interpol_vdep_nests
                                    --> interpol_wind --> sigma
                                    --> interpol_wind_nests --> sigma
                                    --> windalign
                                    --> cll2xy (various projection routines in cmapf1.0.f)
                                    --> cxy2ll
                                    --> interpol_wind_short
                                    --> interpol_wind_short_nests
                        --> calcfluxes
                        --> drydepokernel
                        --> drydepokernel_nest
                        --> partoutput --> caldate

------------------------------------------------------------------------------
%1
 --> readwind --> pbopen
              --> pbgrib
              (--> swap32)
              --> gribex
              --> pbclose
              --> shift_field_0
              --> shift_field
              --> pbl_profile --> psim
                              --> psih
 --> readwind_nests --> pbopen
                    --> pbgrib
                    --> swap32
                    --> gribex
                    --> pbclose
              --> pbl_profile --> psim
                              --> psih
 --> calcpar --> scalev --> ew
             --> obukhov
             --> richardson --> qvsat
             --> getvdep --> caldate
                         --> getrb
```

```
                        --> raerod --> psih
                        --> getrc
                        --> partdep
            --> calcpv
 --> calcpar_nests --> scalev --> ew
            --> obukhov
            --> richardson --> qvsat
            --> getvdep --> caldate
                        --> getrb
                        --> raerod --> psih
                        --> getrc
                        --> partdep
            --> calcpv_nests
 --> verttransform --> cc2gll (cmapf1.0.f)
 --> verttransform_nests --> cc2gll (cmapf1.0.f)
-------------------------------------------------------------------------------
```