The Lagrangian particle dispersion model FLEXPART version 10.4

Ignacio Pisso, Espen Sollum, Henrik Grythe, Nina Kristiansen, Massimo Cassiani, Sabine Eckhardt, Delia Arnold, Don Morton, Rona Thompson, Christine Groot Zwaaftink, et al.

Geosci. Model Dev., 12, 4955–4997, 2019
https://doi.org/10.5194/gmd-12-4955-2019

Geoscientific Model Development
Open Access
EGU

The Lagrangian particle dispersion model FLEXPART version 10.4

Ignacio Pisso[1], Espen Sollum[1], Henrik Grythe[1], Nina I. Kristiansen[1,a], Massimo Cassiani[1], Sabine Eckhardt[1], Delia Arnold[2,3], Don Morton[4], Rona L. Thompson[1], Christine D. Groot Zwaaftink[1], Nikolaos Evangeliou[1], Harald Sodemann[5], Leopold Haimberger[6], Stephan Henne[7], Dominik Brunner[7], John F. Burkhart[8], Anne Fouilloux[8], Jerome Brioude[9], Anne Philipp[6,10], Petra Seibert[11], and Andreas Stohl[1]

[1]Norwegian Institute for Air Research (NILU), Kjeller, Norway
[2]Central Institute for Meteorology and Geodynamics (ZAMG), Vienna, Austria
[3]Arnold Scientific Consulting, Manresa, Spain
[4]Boreal Scientific Computing, Fairbanks, Alaska, USA
[5]Geophysical Institute, University of Bergen and Bjerknes Centre for Climate Research, Bergen, Norway
[6]Department of Meteorology and Geophysics, University of Vienna, Vienna, Austria
[7]Empa, Swiss Federal Laboratories for Materials Science and Technology, Dübendorf, Switzerland
[8]Department of Geosciences, University of Oslo, Oslo, Norway
[9]Laboratoire de l'Atmosphère et des Cyclones (LACy), UMR8105, Université de la Réunion – CNRS – Météo-France, Saint-Denis de La Réunion, France
[10]Aerosol Physics & Environmental Physics, University of Vienna, Vienna, Austria
[11]Institute of Meteorology, University of Natural Resources and Life Sciences, Vienna, Austria
[a]now at: Met Office, FitzRoy Road, Exeter, EX1 3PB, UK

Correspondence: Ignacio Pisso (ip@nilu.no)

Received: 21 December 2018 – Discussion started: 28 January 2019
Revised: 25 July 2019 – Accepted: 7 August 2019 – Published: 2 December 2019

Abstract. The Lagrangian particle dispersion model FLEXPART in its original version in the mid-1990s was designed for calculating the long-range and mesoscale dispersion of hazardous substances from point sources, such as those released after an accident in a nuclear power plant. Over the past decades, the model has evolved into a comprehensive tool for multi-scale atmospheric transport modeling and analysis and has attracted a global user community. Its application fields have been extended to a large range of atmospheric gases and aerosols, e.g., greenhouse gases, short-lived climate forcers like black carbon and volcanic ash, and it has also been used to study the atmospheric branch of the water cycle. Given suitable meteorological input data, it can be used for scales from dozens of meters to global. In particular, inverse modeling based on source–receptor relationships from FLEXPART has become widely used. In this paper, we present FLEXPART version 10.4, which works with meteorological input data from the European Centre for Medium-Range Weather Forecasts (ECMWF) Integrated Forecast System (IFS) and data from the United States National Centers of Environmental Prediction (NCEP) Global Forecast System (GFS). Since the last publication of a detailed FLEXPART description (version 6.2), the model has been improved in different aspects such as performance, physicochemical parameterizations, input/output formats, and available preprocessing and post-processing software. The model code has also been parallelized using the Message Passing Interface (MPI). We demonstrate that the model scales well up to using 256 processors, with a parallel efficiency greater than 75 % for up to 64 processes on multiple nodes in runs with very large numbers of particles. The deviation from 100 % efficiency is almost entirely due to the remaining non-parallelized parts of the code, suggesting large potential for further speedup. A new turbulence scheme for the convective boundary layer has been developed that considers the skewness in the vertical velocity distribution (updrafts and downdrafts) and vertical gradients in air density. FLEXPART is the only model available considering both effects, making it highly accurate for small-scale applications, e.g., to quantify dispersion in the vicinity of a point source. The wet deposi-

tion scheme for aerosols has been completely rewritten and a new, more detailed gravitational settling parameterization for aerosols has also been implemented. FLEXPART has had the option of running backward in time from atmospheric concentrations at receptor locations for many years, but this has now been extended to also work for deposition values and may become useful, for instance, for the interpretation of ice core measurements. To our knowledge, to date FLEXPART is the only model with that capability. Furthermore, the temporal variation and temperature dependence of chemical reactions with the OH radical have been included, allowing for more accurate simulations for species with intermediate lifetimes against the reaction with OH, such as ethane. Finally, user settings can now be specified in a more flexible namelist format, and output files can be produced in NetCDF format instead of FLEXPART's customary binary format. In this paper, we describe these new developments. Moreover, we present some tools for the preparation of the meteorological input data and for processing FLEXPART output data, and we briefly report on alternative FLEXPART versions.

1 Introduction

Multi-scale offline Lagrangian particle dispersion models (LPDMs) are versatile tools for simulating the transport and turbulent mixing of gases and aerosols in the atmosphere. Examples of such models are the Numerical Atmospheric-dispersion Modelling Environment (NAME) (Jones et al., 2007), the Stochastic Time-Inverted Lagrangian Transport (STILT) model (Lin et al., 2003), the Hybrid Single-Particle Lagrangian Integrated Trajectory (HYSPLIT) model (Stein et al., 2015) and the FLEXible PARTicle (FLEXPART) model (Stohl et al., 1998, 2005). LPDMs are stochastic models that compute trajectories for a large number of notional particles that do not represent real aerosol particles but points moving with the ambient flow. The trajectories represent the transport by mean flow as well as turbulent, diffusive transport by unresolved parameterized subgrid-scale transport processes (e.g., turbulence, meandering, deep convection, etc.) and can also include gravitational settling. Each particle carries a certain mass, which can be affected by loss processes such as radioactive decay, chemical loss, or dry and wet deposition.

The theoretical basis for most currently used atmospheric particle models was laid down by Thomson (1987). He introduced the criterion to formulate Lagrangian stochastic models that produce particle trajectories consistent with predefined Eulerian probability density functions in physical and velocity space. Rodean (1996) and Wilson and Sawford (1996) provided detailed descriptions of the theory and formulation of LPDMs in constant density flows and under different atmospheric stability conditions. Stohl and Thomson (1999) extended this to flows with vertically variable air density. An important characteristic of LPDMs is their ability to run backward in time in a framework that is theoretically consistent with both the Eulerian flow field and LPDM forward calculations. This was discussed by Thomson (1987, 1990), further developed by Flesch et al. (1995), and extended to global-scale dispersion by Stohl et al. (2003) and Seibert and Frank (2004). The more practical aspects and efficiency of LPDMs were discussed by Zannetti (1992) and Uliasz (1994). A history of their development was provided by Thomson and Wilson (2013).

Lagrangian models exhibit much less numerical diffusion than Eulerian or semi-Lagrangian models (e.g., Reithmeier and Sausen, 2002; Cassiani et al., 2016), even though some numerical errors also arise in the discretization of their stochastic differential equations (Ramli and Esler, 2016). Due to their low level of numerical diffusion, tracer filaments generated by dispersion in the atmosphere (Ottino, 1989) are much better captured in Lagrangian models than in Eulerian models. It has been noticed, for instance, that Eulerian models have difficulties simulating the fine tracer structures created by intercontinental pollution transport (Rastigejev et al., 2010), while these are well preserved in LPDMs (e.g., Stohl et al., 2003). Furthermore, in Eulerian models a tracer released from a point source is instantaneously mixed within a grid box, whereas Lagrangian models are independent of a computational grid and can account for point or line sources with potentially infinitesimal spatial resolution. When combined with their capability to run backward in time, this means that LPDMs can also be used to investigate the history of air parcels affecting, for instance, an atmospheric measurement site (e.g., for in situ monitoring of atmospheric composition).

The computational efficiency of LPDMs depends on the type of application. One important aspect is that their computational cost does not increase substantially with the number of species transported (excluding aerosol particles with different gravitational settling, for which trajectories deviate from each other), making multispecies simulations efficient. On the other hand, the computational time scales linearly with the number of particles used, while the statistical error in the model output decreases only with the square root of the particle density. Thus, it can be computationally costly to reduce statistical errors, and data input/output can require substantial additional resources. Generally, a high particle density can be achieved with a small number of released particles in the vicinity of a release location, where statistical errors, relative to simulated concentrations, are typically small. However, particle density and thus the relative accuracy of the results decrease with distance from the source. Methods should therefore be used to reduce the statistical error (e.g., Heinz et al., 2003), such as kernels or particle splitting, and it is important to quantify the statistical error.

1.1 The Lagrangian particle dispersion model FLEXPART

One of the most widely used LPDMs is the open-source model FLEXPART, which simulates the transport, diffusion, dry and wet deposition, radioactive decay, and 1st-order chemical reactions (e.g., OH oxidation) of tracers released from point, line, area or volume sources, or filling the whole atmosphere (Stohl et al., 1998, 2005). FLEXPART development started more than 2 decades ago (Stohl et al., 1998) and the model has been free software ever since it was first released. The status as a free software is formally established by releasing the code under the GNU General Public License (GPL) Version 3. However, the last peer-reviewed publication describing FLEXPART (version 6.2) was published as a technical note about 14 years ago (Stohl et al., 2005). Since then, while updates of FLEXPART's source code and a manual were made available from the web page at https://flexpart.eu/ (last access: 30 October 2019), no citable reference was provided. In this paper, we describe FLEXPART developments since Stohl et al. (2005), which led to the current version 10.4 (subsequently abbreviated as v10.4).

FLEXPART can be run either forward or backward in time. For forward simulations, particles are released from one or more sources and concentrations (or mixing ratios) are determined on a regular latitude–longitude–altitude grid. In backward mode, the location where particles are released represents a receptor (e.g., a measurement site). Like in the forward mode, particles are sampled on a latitude–longitude–altitude grid, which in this case corresponds to potential sources. The functional values obtained represent the source–receptor relationship (SRR) (Seibert and Frank, 2004), also called source–receptor sensitivity (Wotawa et al., 2003) or simply emission sensitivity, and are related to the particles' residence time in the output grid cells. Backward modeling is more efficient than forward modeling for calculating SRRs if the number of receptors is smaller than the number of (potential) sources. Seibert and Frank (2004) explained in detail the theory of backward modeling, and Stohl et al. (2003) gave a concrete backward modeling example. FLEXPART can also be used in a domain-filling mode whereby the entire atmosphere is represented by (e.g., a few million) particles of equal mass (Stohl and James, 2004). The number of particles required for domain-filling simulations, not unlike those needed for other types of simulations, depends on the scientific question to be answered. For instance, a few million particles distributed globally are often enough to investigate the statistical properties of air mass transport (e.g., monthly average residence times in a particular area that is not too small) but would not be enough for a case study of airstreams related to a particular synoptic situation (e.g., describing flow in the warm conveyor belt of a particular cyclone).

FLEXPART is an offline model that uses meteorological fields (analyses or forecasts) as input. Such data are available from several different numerical weather prediction (NWP) models. For the model version described here, v10.4, data from the European Centre for Medium-Range Weather Forecasts (ECMWF) Integrated Forecast System (IFS) and data from the United States National Centers of Environmental Prediction (NCEP) Global Forecast System (GFS) can be used. Common spatial resolutions for IFS depending on the application include $1^\circ \times 1^\circ$ at 3 h (standard for older products, e.g., ERA-Interim), $0.5^\circ \times 0.5^\circ$ at 1 h (standard for newer products, e.g., ERA5) and $0.1^\circ \times 0.1^\circ$ at 1 h (current ECMWF operational data). The ECMWF IFS model currently has 137 vertical levels. NCEP GFS input files are usually used at $1^\circ \times 1^\circ$ horizontal resolution, with 64 vertical levels and 3 h time resolution. NCEP GFS input files are also available at $0.5^\circ \times 0.5^\circ$ and $0.25^\circ \times 0.25^\circ$ horizontal resolution. Other FLEXPART model branches have been developed for input data from various limited-area models, for example the Weather Research and Forecasting (WRF) meteorological model (Brioude et al., 2013) and the Consortium for Small-scale Modeling (COSMO) model (Oney, 2015), which extend the applicability of FLEXPART down to the mesogamma scale. Notice that the turbulence parameterizations of FLEXPART are valid at even smaller scales. Another FLEXPART model version, FLEXPART–NorESM/CAM (Cassiani et al., 2016), uses the meteorological output data generated by the Norwegian Earth System Model (NorESM1-M) with its atmospheric component CAM (Community Atmosphere Model). The current paper does not document these other model branches, but most share many features with FLEXPART v10.4 and some are briefly described in Appendix C. A key aspect of these model branches is the ability to read meteorological input other than that from ECMWF or NCEP.

1.2 FLEXPART and its history

FLEXPART's first version (v1) was a further development of the trajectory model FLEXTRA (Stohl et al., 1995) and was coded in Fortran 77. It provided gridded output of concentrations of chemical species and radionuclides. Its meteorological input data were based on the ECMWF-specific GRIB-1 (gridded binary) format. The model was first applied in an extensive validation study using measurements from three large-scale tracer experiments (Stohl et al., 1998). A deposition module was added in version 2. Version 3 saw improvements in performance and the addition of a subgrid-scale terrain effects parameterization. In v3.1 the output format was optimized (sparse matrix) and mixing ratio output could optionally be produced. It also allowed for the output of particle positions. Furthermore, a density correction was added to account for decreasing air density with height in the boundary layer (Stohl and Thomson, 1999). Further v3 releases included the addition of a convection scheme (Seibert et al., 2001) based on Emanuel and Živković-Rothman (1999), the option to calculate mass fluxes across grid cell faces and age spectra, and free format input (v3.2). The preliminary convection scheme of v3.2 was revised in v4 (see

Forster et al., 2007). In v5 the output unit of backward calculations was changed to seconds and improvements in the input/output handling were made. Comprehensive validation of these early FLEXPART versions was done during intercontinental air pollution transport studies at the end of the 1990s and early 2000s (Stohl and Trickl, 1999; Forster et al., 2001; Spichtinger et al., 2001; Stohl et al., 2002, 2003). Special developments were also made in order to extend FLEXPART's forecasting capabilities for large-scale field campaigns (Stohl et al., 2004). Version 6.0 saw corrections to the numerics in the convection scheme, the addition of a domain-filling option used, for instance, in water cycle studies (Stohl and James, 2004) and the possibility to use nested output. Version 6.2, which added the ability to model sources and receptors in both mass and mixing ratio units (Seibert and Frank, 2004), is currently the last version described in a publication (Stohl et al., 2005). A separate sister model branch (v6.4) was adapted to run with NCEP GFS meteorological input data. The current paper describes the most important model developments since v6.2 (for ECMWF) and v6.4 (for GFS).

Version 8.0 unified the model branches based on ECMWF IFS and NCEP GFS input data in one source package but still required the building of two different executables. Importantly, Fortran 90 constructs were introduced in parts of the code, such as initial support for dynamic memory allocation. Furthermore, a global land use inventory was added, allowing for more accurate dry deposition calculations everywhere on the globe (before, land use data were provided only for Europe). The reading of the – at the time – newly introduced GRIB-2 format with the ECMWF grib_api library was implemented in v8.2. An option to calculate the sensitivity to initial conditions in backward model runs (in addition to the emission sensitivity calculations) was also implemented in v8.2. Version 8 was also the first version that distinguished between in-cloud and below-cloud scavenging for washout, relying on simple diagnostics for clouds based on grid-scale relative humidity. With a growing number of parameters defining removal processes, each species was given its own definition file, whereas in previous versions the properties for all species were contained in a single file. The gravitational settling scheme was improved in v8.2.1 (Stohl et al., 2011), and this is briefly described in this paper in section 2.3.

For v9, the code was transformed to the Fortran 90 free-form source format. The option to read the run specifications from Fortran namelists instead of the standard input files was introduced, as described in Sect. 5 of this paper. This change was motivated by the resulting greater flexibility, in particular with regard to setting default values, optional arguments, when new developments require adding new parameters and when specifying parameter lists. In addition, an option to produce output in compressed NetCDF 4 format was provided (see Sect. 6.3). Another option to write some model output only for the first vertical layer to save storage space for inverse modeling applications was also introduced (Thompson and Stohl, 2014) (see Sect. 2.6).

1.3 FLEXPART version 10.4

For v10.4 of FLEXPART, described in this paper, several more changes and improvements were made. First, an optional new scheme applying more realistic skewed rather than Gaussian turbulence statistics in the convective atmospheric boundary layer (ABL) was developed (Sect. 2.1). Second, the wet deposition scheme for aerosols was totally revised (Grythe et al., 2017), introducing dependencies on aerosol size, precipitation type (rain or snow), and distinguishing between in-cloud and below-cloud scavenging (see Sect. 2.4). The code now also allows for the reading of three-dimensional (3-D) cloud water fields from meteorological input files. Third, a method to calculate the sensitivity of deposited quantities to sources in backward mode was developed (Sect. 2.5) Fourth, chemical reactions with the hydroxyl radical (OH) are now made dependent on the temperature and vary sub-monthly (Sect. 2.7). Fifth, large parts of the code were parallelized using the Message Passing Interface (MPI), thus facilitating a substantial speedup for certain applications (see Sect. 3), and the code was unified so that a single executable can now use both ECMWF and GFS input data. Sixth, a dust mobilization scheme that can be run as a FLEXPART preprocessor was developed (Sect. 2.8). Seventh, the software used to retrieve data from the ECMWF has been modernized and can now also be used by scientists from non-ECMWF member states (Sect. 5.2.1). Finally, a testing environment was created that allows users to verify their FLEXPART installation and compare results (Sect. 7).

Despite the many changes and additions, in large part the operation of FLEXPART v10.4 still resembles the original version 1 design. Throughout this paper, we avoid repeating information on aspects of the model that have not changed since earlier model descriptions. The paper should therefore always be considered together with the publications of Stohl et al. (1998, 2005). To provide the necessary context for the rest of this paper, we provide a brief overview of the FLEXPART v10.4 directory structure in Table 1. The source code is contained in directory `src`. The pathnames of the input and output directories are stated in the file `pathnames` read by the FLEXPART executable. The directory `options` contains the parameters that define a run in files such as `COMMAND` (e.g., start and end times of the simulation, output frequency, etc.), `RELEASES` (definition of the particle releases), `OUTGRID` (output grid specifications) and others. All the output is written in a directory unique for each run. There are also other directories containing the model testing environment and example runs, as well as preprocessing and post-processing software (see Table 1).

Sensu stricto FLEXPART consists of the (Fortran) source files required to build an executable, not including external libraries such as those needed for input reading. The make-

files and the sample input as provided in the "options" may also be included under this term. However, in order to do real work with FLEXPART, one also needs to obtain meteorological input data (in the case of the ECMWF this is not trivial, so the `flex_extract` package is provided), and one needs to do post-processing. This is the reason why we include a selection of such tools here.

2 Updates of the model physics and chemistry

This section gives an overview of the main updates of the model physics and chemistry since the last published FLEXPART version, v6.2 (Stohl et al., 2005). Some developments have been published already separately, and in such cases we keep the description short, focusing on technical aspects of the implementation in FLEXPART that are important for model users or demonstrating applications not covered in the original papers.

2.1 Boundary layer turbulence

Subgrid-scale atmospheric motions unresolved by the meteorological input data need to be parameterized in FLEXPART. This is done by adding stochastic fluctuations based on Langevin equations for the particle velocity components (Stohl et al., 2005). In the ABL, the stochastic differential equations are formulated according to the well-mixed criteria proposed by Thomson (1987). Until FLEXPART version 9.2, the Eulerian probability density functions (PDFs) for the three velocity components were assumed to be three independent Gaussian PDFs. However, for the vertical velocity component, the Gaussian turbulence model is well suited only for stable and neutral conditions. In the convective ABL (CBL), turbulence is skewed since a larger area is occupied by downdrafts than by updrafts (e.g., Stull, 1988; Luhar and Britter, 1989). In such conditions, the Gaussian turbulence model is not appropriate for sources within the ABL, as it cannot reproduce the observed upward bending of plumes from near-ground sources or the rapid downward transport of plumes from elevated sources (Venkatram and Wyngaard, 1988). However, the Gaussian approximation has negligible influence once the tracer is mixed throughout the whole ABL.

Cassiani et al. (2015) developed an alternative Langevin equation model for the vertical particle velocity including both skewed turbulence and a vertical density gradient, which is now implemented in FLEXPART v10.4. This scheme can be activated by setting the switch `CBL` to 1 in the file `COMMAND`. In this case, the time step requirement for numerical stability is much more stringent than for the standard Gaussian turbulence model (typically, values of `CTL=10` and `IFINE=10` are required, also set in the file `COMMAND`). Therefore, also considering that the computation time required for each time step is about 2.5 times that of

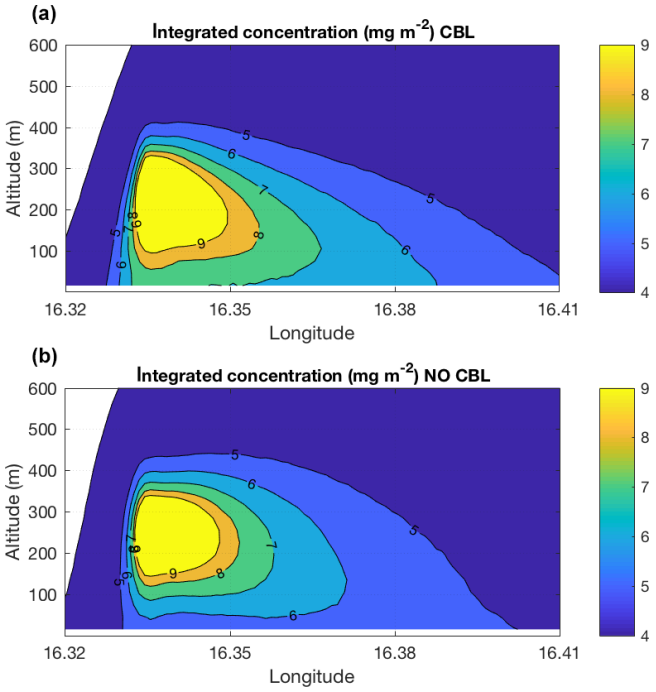

Figure 1. Comparison of FLEXPART results obtained with the skewed turbulence parameterization **(a)** and with the Gaussian turbulence parameterization **(b)**. Shown are the tracer concentrations integrated over all latitudes as a function of altitude and longitude. The simulations used a point source emitting 100 kg of tracer per hour for a period starting on 1 July 2017 at 12:00 UTC and ending at 13:30 UTC. The source was located near Vienna (Austria) at 47.9157° N and 16.3274° E, 250 m above ground level. Results are averaged for the time period 12:40 to 13:00 UTC. Notice that the maximum ground-level concentration in panel **(a)** (ca. 7 mg m^{-2}) is about 30 % higher than in panel **(b)** (5 mg m^{-2}).

the standard Gaussian formulation, the CBL option is much more computationally demanding and not recommended for large-scale applications. However, for studies of tracer dispersion in the vicinity of individual point sources, the CBL option is essential to reproduce the characteristic features of CBL dispersion (Weil, 1985), while the additional computational burden remains tolerable.

Figure 1 shows a comparison between two simulations of dispersion from an elevated source, with the skewed and with the Gaussian turbulence model. It can be seen that the maximum time-averaged ground-level concentration is about 30 % higher for the skewed turbulence parameterization. This is the result of the plume centerline tilting downward to the surface in the vicinity of the source for the skewed turbulence case due to downdrafts being more frequent than updrafts. The plume also spreads faster in this case. These results are similar to those obtained by others (e.g., Luhar and Britter, 1989).

It is important to note that the CBL formulation smoothly transits to a standard Gaussian formulation when the stabil-

Table 1. Directory structure overview of the FLEXPART v10.4 software distribution. All listed directories are subdirectories of the installation root directory `$flexhome/`.

	Subdirectory or file	Contents	Description and comments

`$flexhome/`	`pathnames` (file)	`$options/`
		`$output/`
		`$flex_winds/`
		`$AVAILABLE`
			path to `options` directory
			path to `output` directory
			path to meteorological input files
			path to `AVAILABLE` file

	`src/`	`*.f90`
		`makefile`
		`FLEXPART`
			Fortran source files
			see Sect. 4 and Appendix A
			executable file (see Sect. 4)

	`options/`	COMMAND, RELEASES, OUTGRID, SPECIES, AGECLASSES, OUTGRID_NEST, RECEPTORS, IGBP_int1.dat, surfdata.t, surfdepo.t, OH_variables.bin	user input files see Sect. 5 and Table 7
	`AVAILABLE`	list of meteorological input data files	file containing list, see Sect. 5
	`output/`	FLEXPART output files	see Sect. 6 and Table 11
	`preprocess/`	`flex_extract/`	see Sect. 5.2
	`postprocess/`	`read_flex_fortran/`, `read_flex_matlab/`,	see Sect. 6.4
	`tests/` `tests/examples/`	development tests for FLEXPART and ancillary software example runs illustrating various FLEXPART functionalities	see Sect. 7 and Appendix C

ity changes towards neutral (Cassiani et al., 2015). However, the actual equation solved inside the model for the Gaussian condition is still different from the standard version as actual particle velocities rather than the scaled ones are advanced (see, e.g., Wilson et al., 1981; Rodean, 1996). Full details of the CBL implementation can be found in Cassiani et al. (2015).

To date, FLEXPART has mainly been used for large-scale applications. With this new CBL option, FLEXPART is now also well suited for the simulation of small-scale tracer dispersion or for the inverse modeling of point source emissions from near-field measurements – at least if the resolved winds are representative of the situation considered. In fact, to our knowledge FLEXPART is the only particle model considering both skewness in the vertical velocity distribution and vertical gradients in air density. Both these effects are particularly important in deep CBLs and can be additive with respect to simulated ground-level concentrations.

2.2 Turbulent diffusivity above the boundary layer

Above the atmospheric boundary layer, turbulent fluctuations can be represented with a turbulent diffusivity. The value of the diffusivity tensor controls the size and lifetimes of the filamentary structures caused by advection. Diffusivities are converted into velocity scales using $\sigma_{v_i} = \sqrt{\frac{2D_i}{\mathrm{d}t}}$, where i is the direction. This corresponds to a mean diffusive displacement of $\sigma_{x_i} = \sqrt{2D_i \mathrm{d}t}$, characteristic of Brownian motion. For i, only the vertical (v) and horizontal (h) directions are considered. The value of the vertical diffusivity D_z is related to the value of the horizontal diffusivity D_h by the square of the typical atmospheric aspect ratio for tracer structures $\kappa \approx 100$–250 (Haynes and Anglade, 1997).

FLEXPART uses by default a constant vertical diffusivity $D_z = 0.1\,\mathrm{m^2\,s^{-1}}$ in the stratosphere, following Legras et al. (2003), whereas a horizontal diffusivity $D_\mathrm{h} = 50\,\mathrm{m^2\,s^{-1}}$ is used in the free troposphere. In general in the atmosphere, the values of the turbulent diffusivity depend on time and location, showing in particular seasonal, latitudinal and altitude variability: e.g., $D_\mathrm{v} = 10^{-2}\,\mathrm{m^2\,s^{-1}}$ in the stratosphere (Balluch and Haynes, 1997) and $D_\mathrm{h} = 10^4\,\mathrm{m^2\,s^{-1}}$ (Pisso et al., 2009) in the troposphere. The values can be modified by the user in the `COMMAND` file (namelist variables `d_trop` and `d_strat`, in $\mathrm{m^2\,s^{-1}}$). As mentioned above, $D_\mathrm{h} \approx \kappa^2 D_z$, and therefore both values can be used interchangeably.

In FLEXPART version 6.2, the stratosphere and troposphere were distinguished based on a threshold of 2 pvu (potential vorticity units), with a maximal height of 18 km in the tropics and a minimal height of 5 km elsewhere. Such a threshold is well suited to midlatitudes, but it can differ from the thermal tropopause in the polar regions and close to the Equator. In FLEXPART 10.4, the thermal tropopause definition is used and is calculated using the lapse rate definition (Hoinka, 1997).

2.3 Gravitational settling

Gravitational settling of aerosols is implemented in FLEXPART as a process that changes the particle trajectories. The settling velocity is determined at each time step and added

to the vertical wind velocity. In simulations in which a particle represents several species, all species are transported with the settling velocity of the first species. If this is not intended, simulations for the different species must be run separately. Gravitational settling velocities are also used in the calculation of dry deposition.

In older FLEXPART versions, gravitational settling was calculated using a single dynamic viscosity of air. With FLEXPART 8.2.1, the gravitational settling calculation was generalized to higher Reynolds numbers and it takes into account the temperature dependence of dynamic viscosity. This is done in subroutine `get_settling.f90` in an iterative loop, wherein both the Reynolds number and the settling velocity are determined (Naeslund and Thaning, 1991). For initialization of the loop, Stokes' law and a constant viscosity estimate is used. The dynamic viscosity is calculated as a function of temperature using the formula of Sutherland (1893). A spherical shape of the particles is assumed in the settling calculation, which could be further extended in the future to allow for more complex particle shapes. For particle sizes of about 10 µm, settling velocities in the new FLEXPART version are not much different from earlier versions using the old settling calculation, typically less than 20 %. However, the differences are largest in the cold upper troposphere, implying, for instance, changes in the residence time of volcanic ash particles at heights relevant for aviation. The residence times in the upper troposphere are increased with the new scheme, but the effect is not particularly large, typically on the order of 20 %.

2.4 Wet deposition

In FLEXPART, the calculation of wet scavenging is divided into three parts. First, where scavenging occurs and which form it takes is determined (e.g., below- or within-cloud scavenging). Second, the scavenging coefficient is determined. Third, the actual removal of particle mass is calculated.

With respect to the first part, it is important to understand that wet scavenging occurs only in the presence of clouds and where precipitation occurs. In air columns without clouds, above the top of the clouds, and where neither the large-scale nor the convective precipitation rate exceeds $0.01\,\mathrm{mm\,h^{-1}}$, no scavenging occurs. To quickly know where a particle is located relative to the clouds, in subroutines `verttransform_ecmwf.f90` and`verttransform_gfs.f90` each grid cell is categorized as being in a cloud-free column, above a cloud, inside a cloud or below a cloud. This cloud definition has been completely revised compared to earlier versions and is described in Sect. 2.4.1.

With respect to the second step, the scavenging coefficient Λ ($\mathrm{s^{-1}}$) is determined in subroutine `get_wetscav.f90`. After a series of updates, in particular Grythe et al. (2017), FLEXPART now distinguishes between below-cloud and in-cloud scavenging and also has different parameterizations of Λ for gases and particles. For the latter, it also distinguishes between liquid-phase and ice-phase states. This yields in total six different parameterizations for Λ, described in Sect. 2.4.2 and 2.4.3.

In the third step, the removal of particle mass due to wet deposition is calculated. It takes the form of an exponential decay process (McMahon, 1979),

$$m(t+\Delta t) = m(t)\exp(-\Lambda\,\Delta t)\ , \tag{1}$$

where m is the particle mass (kg) (it can also be a mass mixing ratio, depending on settings in file `COMMAND`). This removal of particle mass and corresponding accumulation of deposition at the surface is calculated in subroutine `wetdepo.f90` and has not been changed since earlier versions.

2.4.1 Definition of clouds, cloud water content and precipitation

The location of clouds, the total cloud column water content and phase, and precipitation intensity and phase are needed in the calculation of the wet scavenging. Therefore, a three-dimensional cloud mask is defined in subroutine `verttransform_ecmwf.f90` (or `verttransform_gfs.f90` for GFS data). In previous FLEXPART versions, the cloud definition scheme was very simple and based on relative humidity (RH). In grid columns with precipitation, grid cells with RH > 80 % were defined as in-cloud, and those with RH < 80 % were set as below-cloud up to the bottom of the uppermost layer with RH > 80 %. This was appropriate for the first version of FLEXPART, as the ECMWF had a similarly simple definition of clouds and more detailed information was not available from the ECMWF archives at the time.

If no cloud information is available from the meteorological data, the old RH-based scheme is still used in FLEXPART. However, nowadays, specific cloud liquid water content (CLWC; $\mathrm{kg\,kg^{-1}}$) and specific cloud ice water content (CIWC; $\mathrm{kg\,kg^{-1}}$) are available as 3-D fields in meteorological analyses from the ECMWF, and NCEP also provides the 3-D cloud water (liquid plus ice) mixing ratio (CLWMR; $\mathrm{kg\,kg^{-1}}$), from here on referred to as q_c. A cloudy grid cell is defined when $q_\mathrm{c} > 0$. FLEXPART v10.4 can ingest the ECMWF CLWC and CIWC either separately or as a sum (q_c =CLWC+CIWC). However, to save storage space, we recommend retrieving only the sum, q_c, from the ECMWF, as the relative fractions of ice and liquid water can be parameterized quite accurately using Eq. (4).

The column cloud water (c_l; $\mathrm{kg\,m^{-2}}$), which is needed for the in-cloud scavenging parameterization, is calculated by integrating q_c over all vertical z levels:

$$c_\mathrm{l} = \sum_z q_\mathrm{c}(z)\,\rho_\mathrm{air}(z)\,\Delta z\ , \tag{2}$$

where $\rho_{\mathrm{air}}(z)$ is the density of the air in the grid cell, and Δz is the vertical extent of the grid cell. In older FLEXPART versions, c_{l} was parameterized based on an empirical equation given in Hertel et al. (1995) using the subgrid (see below for a description of how subgrid is defined) surface precipitation rate I_{s} ($\mathrm{mm\,h^{-1}}$). While such a parameterization is not needed anymore if q_{c} is available, it is still activated in the case that cloud water input data are missing. However, in order to ensure that c_{l} from the parameterization is statistically consistent with the cloud data, we derived the modified expression

$$c_{\mathrm{l}} = 0.5 \times I_{\mathrm{s}}^{0.36} \tag{3}$$

using a regression analysis between existing cloud and precipitation data.

Precipitation is not uniform within a grid cell. To account for subgrid variability, it is assumed that precipitation is enhanced within a subgrid area and that no precipitation (and thus no scavenging) occurs outside this subgrid area. The subgrid area fraction and precipitation rate (I_{s}) are estimated from the grid-scale precipitation rate (I_t) based on values tabulated in Hertel et al. (1995). This parameterization of subgrid variability is used for all scavenging processes in FLEXPART and maintained from previous FLEXPART versions as described in Stohl et al. (2005).

The precipitation phase needed for the below-cloud scavenging scheme is simply based on ambient grid-scale temperature, with snow occurring below 0 °C and rain above. For cloud water, c_{l}, we assume a temperature-dependent mixture of liquid and solid particles, for which the liquid fraction (α_{L}) is calculated based on the local temperature T,

$$\alpha_{\mathrm{L}} = \left(\frac{T - T_{\mathrm{I}}}{T_{\mathrm{L}} - T_{\mathrm{I}}}\right)^2 , \tag{4}$$

where $T_{\mathrm{L}} = 0°\mathrm{C}$ and $T_{\mathrm{I}} = -20\,°\mathrm{C}$. For $T > T_{\mathrm{L}}$, $\alpha_{\mathrm{L}} = 1$ and for $T < T_{\mathrm{I}}$, $\alpha_{\mathrm{L}} = 0$. Even when CLWC and CIWC are available as separate fields, we derive the liquid fraction (α_{L}) of cloud water from the local temperature. The ice fraction α_{I} is $1 - \alpha_{\mathrm{L}}$. Comparisons have shown that CLWC is very accurately reproduced by $\alpha_{\mathrm{L}} q_{\mathrm{c}}$.

The cloud information should be linearly interpolated like the other variables, and situations in which the diagnosed cloud is incompatible with the precipitation rate (be it because of interpolation or because of convective precipitation accompanied by too shallow or lacking grid-scale clouds) need to receive special treatment. This is planned for a version upgrade in the near future in conjunction with a better interpolation scheme for precipitation (see Hittmeir et al., 2018). In certain cases, the deposition calculation of FLEXPART might be improved by using higher-resolution precipitation data from other sources such as from radar observations (Arnold et al., 2015); however, as the precipitation and ECMWF cloud data, and also the precipitation and wind fields, may not match, this does not guarantee better results.

2.4.2 Below-cloud scavenging

For gases, the scavenging coefficient, Λ, for below-cloud scavenging is calculated as described in Asman (1995),

$$\Lambda = A I_{\mathrm{s}}^{B} , \tag{5}$$

where the scavenging parameters A and B depend on the chemical properties of the gas and are specified in the `SPECIES_nnn` file as described in Sect. 5.1.3 (`nnn` represents the species number (0–999) used for the simulation). In older FLEXPART versions, this scheme was used also for aerosols; however, Grythe et al. (2017) developed a new aerosol scavenging scheme that is implemented in FLEXPART v10.4 and briefly summarized here.

The relevant processes of collision and attachment of ambient aerosol particles to falling precipitation depend mainly on the relationship between the aerosol and hydrometeor size and type (rain or snow) as well as to a lesser degree on the density and hygroscopicity of the aerosol. In FLEXPART v10.4, the dependence of scavenging on the sizes of both the aerosol and falling hydrometeors are taken into account by the schemes of Laakso et al. (2003) for rain and Kyrö et al. (2009) for snow. Both schemes follow the equation

$$\begin{aligned}\log_{10}\left(\frac{\Lambda}{\Lambda_0}\right) = C_*\big(&a + b\,D_{\mathrm{p}}^{-4} + c\,D_{\mathrm{p}}^{-3} + d\,D_{\mathrm{p}}^{-2} \\ &+ e\,D_{\mathrm{p}}^{-1} + f\left(\frac{I_{\mathrm{s}}}{I_0}\right)^{0.5}\big) ,\end{aligned} \tag{6}$$

where $D_{\mathrm{p}} = \log_{10}\frac{d_{\mathrm{p}}}{d_{\mathrm{p0}}}$, d_{p} is the particle dry diameter provided in the `SPECIES_nnn` file, $d_{\mathrm{p0}} = 1\,\mathrm{m}$, $\Lambda_0 = 1\,\mathrm{s^{-1}}$ and $I_0 = 1\,\mathrm{mm\,h^{-1}}$. Coefficients for factors a–f are different for rain and snow scavenging and are given in Table 2. The C_* values are collection efficiencies that reflect the properties of the aerosol and must be given for both rain ($C_* = C_{\mathrm{rain}}$) and snow scavenging ($C_* = C_{\mathrm{snow}}$) in the `SPECIES_nnn` file. Notice that by setting $C_{\mathrm{snow}} = 0$, below-cloud scavenging by snowfall is switched off (similarly, $C_{\mathrm{rain}} = 0$ for rain).

2.4.3 In-cloud scavenging

For in-cloud scavenging of both aerosols and gases, Λ is calculated as described in Grythe et al. (2017):

$$\Lambda = i_{\mathrm{c_r}} S_i I_{\mathrm{s}} , \tag{7}$$

where $i_{\mathrm{c_r}} = 6.2$ is the cloud water replenishment factor, which was determined empirically in Grythe et al. (2017) (there it was named ic_{r}), and S_i is proportional to the in-cloud scavenging ratio, which is derived differently for gases and aerosols.

For gases, $S_i = \frac{1}{\frac{1-c_{\mathrm{l}}}{HRT} + c_{\mathrm{l}}}$, where H is Henry's constant (describing the solubility of the gas and specified in the `SPECIES_nnn` file), R is the gas constant and T is temperature. Notice that this is applied for both liquid-phase and ice clouds.

Table 2. Parameters used in Eq. (6) for below-cloud scavenging.

	C_*	a	b	c	d	e	f	Reference
Rain scavenging	C_{rain}	274.36	332 839.6	226656	58 005.9	6588.38	0.24498	Laakso et al. (2003)
Snow scavenging	C_{snow}	22.7	0	0	1321	381	0	Kyrö et al. (2009)

For aerosols, the in-cloud scavenging is dominated by activated particles forming cloud droplets or ice nuclei. Those may eventually combine to form a hydrometeor that falls out of the cloud, thus removing all aerosol particles contained in it. Therefore, S_i depends on the nucleation efficiency (F_{nuc}) and c_{l}:

$$S_i = \frac{F_{\text{nuc}}}{c_{\text{l}}} \,. \tag{8}$$

F_{nuc} describes how efficiently the aerosol particles are activated as cloud droplet condensation nuclei (CCN) or ice nuclei (IN):

$$F_{\text{nuc}} = \alpha_{\text{L}} \text{CCN}_{\text{eff}} + \alpha_{\text{I}} \text{IN}_{\text{eff}} \,, \tag{9}$$

where the relative abundances of the liquid and ice phase are accounted for by the factor α_{L}. Values for the efficiencies, CCN_{eff} and IN_{eff}, are available from the literature for many different types of aerosols (e.g., black carbon, mineral dust particles or soluble particles) and some have been collected in `SPECIES_nnn` files distributed with FLEXPART (see Sect. 5.1.3). The CCN_{eff} and IN_{eff} values are set for an aerodynamic particle radius of 1 µm, but CCN and IN efficiencies increase with increasing particle size. The in-cloud parameterization takes this into account. For further details on the wet scavenging scheme used in FLEXPART, see Grythe et al. (2017).

2.4.4 Influence of wet scavenging on the aerosol lifetime

Aerosol wet scavenging controls the lifetime of most aerosols. In Fig. 2, we compare modeled *e*-folding lifetimes from a number of FLEXPART simulations using different model versions and switching off in-cloud and below-cloud scavenging in FLEXPART v10.4 with measured lifetimes. The parameter settings in FLEXPART used for these comparisons were the same as used by Grythe et al. (2017). To derive aerosol lifetimes in a consistent way from both measurements and model simulations, a radionuclide attached to ambient aerosol and a noble gas radionuclide were used. Kristiansen et al. (2016) used the same method to compare many different aerosol models, and we refer to their paper for more details on the method. For our model simulations, several size bins of aerosols were used, though total concentrations and lifetimes are largely controlled by 0.4 and 0.6 µm particles (see Grythe et al., 2017). *E*-folding lifetimes increase from 5.8 to 10.0 d between FLEXPART v9 and v10.4. A simulation performed with v10.4 but which emulated the in-cloud scavenging of v9 showed that the difference is mainly due to the decreased in-cloud scavenging in the new removal scheme compared to the old one. Notice that the lifetime obtained with v10.4 is much closer to the observation-based lifetimes. Turning off the below-cloud removal has a relatively small effect, increasing the lifetime to 11 d, whereas turning off the in-cloud removal extends the lifetime to the unrealistic value of 66 d (see bottom two panels in Fig. 2). This highlights the dominant role of in-cloud removal for accumulation-mode particles in FLEXPART.

Notice that compared to older versions of FLEXPART, the `SPECIES_nnn` files now include additional parameters related to the wet deposition scheme. Old input files, therefore, need to be updated for use with FLEXPART v10.4. The required format changes are detailed in Sect. 5.1.3.

2.5 Source–receptor matrix calculation of deposited mass backward in time

When running FLEXPART forward in time for a depositing species with a given emission flux (kilograms per release as specified in file `RELEASES`), the accumulated wet and dry deposition fluxes (ng m^{-2}) are appended to the FLEXPART output files (`grid_conc_date` and/or `grid_pptv_date`, for which `date` represents the date and time in format `YYYYMMDDhhmmss`; see Sect. 6) containing the atmospheric concentration and/or volume mixing ratio output. The deposition is always given in mass units, even if atmospheric values are given in mixing ratio units. In contrast to concentration values, deposition quantities are accumulated over the time of the simulation, so the deposited quantities generally increase during a simulation (except when radioactive decay is activated, which also affects deposited quantities and can decrease them).

As discussed in Sect. 1, running FLEXPART backward in time for calculating SRRs is more efficient than running it forward if the number of (potential) sources is larger than the number of receptors. For atmospheric concentrations (or mixing ratios), the backward mode has been available from the very beginning and in an improved form since FLEXPART v5 (Stohl et al., 2003; Seibert and Frank, 2004). This has proven very useful for the interpretation of ground-based, shipborne or airborne observations (e.g., to characterize sources contributing to pollution plumes). Furthermore, the inversion scheme FLEXINVERT (Thompson and Stohl, 2014) that is used to determine the fluxes of greenhouse gases

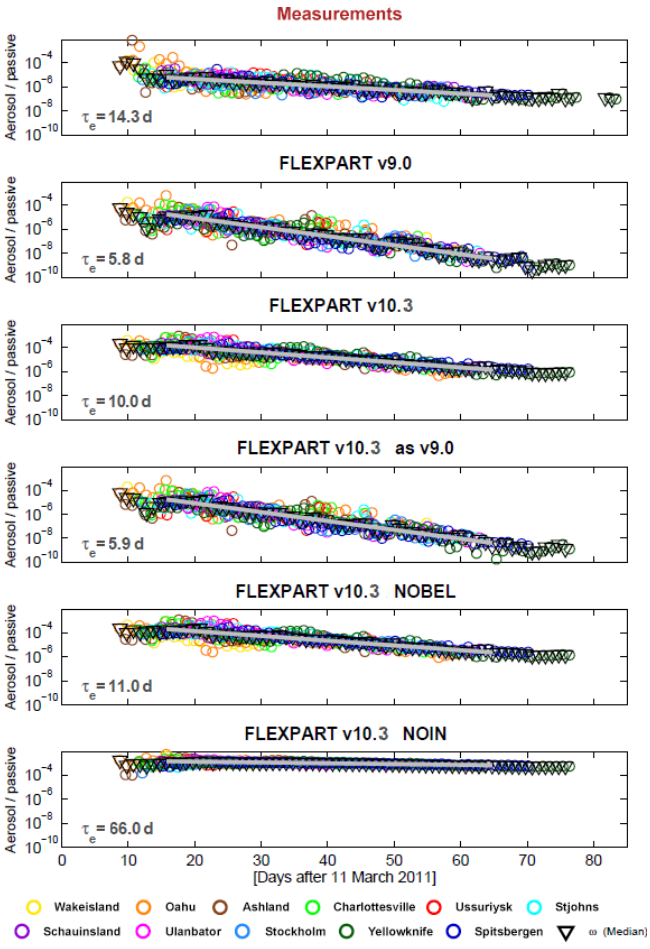

Figure 2. Aerosol lifetimes estimated from the decrease in radionuclide ratios (aerosol-bound ^{137}Cs and noble gas ^{133}Xe as a passive tracer) with time after the Fukushima nuclear accident, as measured and modeled at a number of global measurement stations. For details on the method, see Kristiansen et al. (2016). *E*-folding lifetimes, τ_e, are estimated based on fits to the data and reported in each panel. In the top panel, the observed values are shown and in subsequent panels from the top, modeled values are given for FLEXPART v9, FLEXPART v10.4, FLEXPART v10.4 with parameter settings to emulate removal as in v9, FLEXPART v10.4 with no below-cloud removal and FLEXPART v10.4 with no in-cloud removal.

is based on backward simulations. However, there are also measurements of deposition on the ground, e.g., in precipitation samples or ice cores, and for this type of measurement no backward simulations were possible until recently. Therefore, Eckhardt et al. (2017) introduced the option to calculate SRR values in backward mode also for wet and dry deposition, and a first application to ice core data was presented by McConnell et al. (2018). It is anticipated that quantitative interpretation of ice core data will be a major application of the new backward mode, which is efficient enough to allow for the calculation of, for example, 100 years of seasonally resolved deposition data in less than 24 h of computation time.

We illustrate the different backward modes and explain the required settings with an example. The calculations were run for a single receptor location, Ny-Ålesund in Spitsbergen (78.93° N, 11.92° E) and for the 24 h period from 18 August 2013 at 20:00 UTC to 19 August 2013 at 20:00 UTC. SRR values are calculated for the atmospheric concentration averaged over the layer 0–100 m a.g.l., as well as for wet and dry deposition. The substance transported is black carbon (BC), which is subject to both dry and wet deposition. Backward simulations for wet and dry deposition must always be run separately. In order to obtain SRR values for total deposition, results for wet and dry deposition need to be summed.

The backward mode is activated by setting the simulation direction, `LDIRECT` in file `COMMAND` (see Sect. 5), to −1. The three simulations are obtained by consecutively setting `IND_RECEPTOR` to 1 (for concentration), 3 (wet deposition) and 4 (dry deposition). `IND_SOURCE` is always set to 1, meaning that the sensitivities (SRR values) are calculated with respect to physical emissions in mass units. A complete list of possible options is reported in Table 1 of Eckhardt et al. (2017).

Figure 3 shows the resulting SRR (i.e., emission sensitivity) fields for the concentration as well as dry and wet deposition at the receptor. Dry deposition occurs on the Earth's surface, and therefore particles are released in a shallow layer adjacent to the surface. Its height is consistent with the shallow depth over which dry deposition is calculated in forward mode (user settings for the release height are ignored for dry deposition backward calculations). Dry deposition rates are the product of the surface concentration and the deposition velocity. Therefore, the SRR fields for surface concentration (Fig. 3a) and dry deposition (Fig. 3b) show similar patterns, in this case indicating high sensitivity for sources over Scandinavia and northwestern Russia. The differences in the spatial patterns are mainly due to temporal variability in the dry deposition velocity at the receptor caused by varying meteorological conditions (e.g., stability) and surface conditions during the 24 h release interval.

Wet deposition, on the other hand, can occur anywhere in the atmospheric column from the surface to the top of the precipitating cloud. FLEXPART automatically releases particles in the whole atmospheric column (again, user settings for the release height are ignored), but particles for which no scavenging occurs (e.g., those above the cloud top or when no precipitation occurs) are immediately terminated. Therefore, and because of the vertical variability of the scavenging process, the sensitivity for the deposited mass can deviate significantly from the sensitivity corresponding to surface concentration. Here (Fig. 3c), the sensitivity is high over Scandinavia and northwestern Russia, as was already seen for surface air concentrations and dry deposition. However, in addition, sources located in North America and eastern

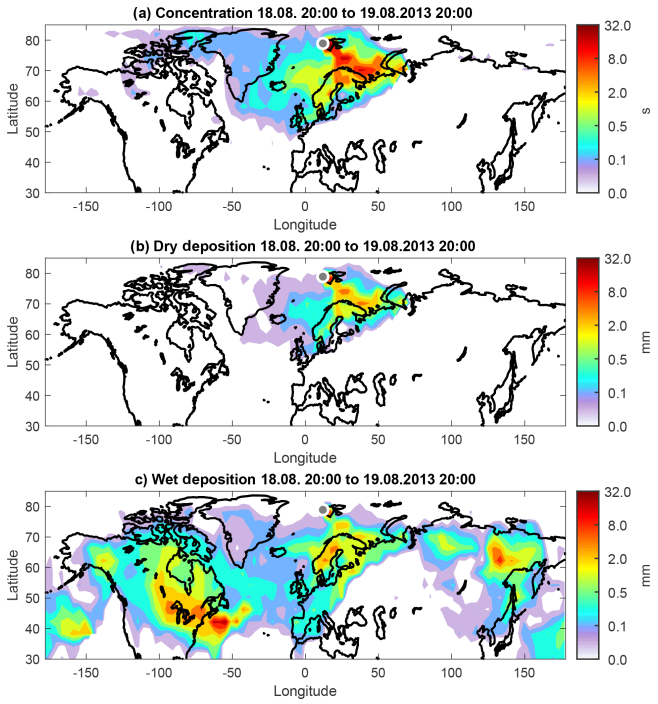

Figure 3. Source–receptor relationships (for emissions occurring in the lowest 100 m a.g.l.) for black carbon observed at Ny-Ålesund in Svalbard for a 24 h period starting on 18 August 2013 at 20:00 UTC. The sensitivities were calculated for **(a)** concentrations (s) in the layer 0–100 m a.g.l., **(b)** dry deposition (mm) and **(c)** wet deposition (mm).

Siberia also contribute strongly to wet deposition. The maximum over the ocean close to the North American east coast is likely due to lifting in a warm conveyor belt, followed by fast transport at high altitude.

Concentration, dry deposition and wet deposition at the receptor can be calculated from the SRR fields shown in Fig. 3 as follows.

$$\begin{aligned} c &= \boldsymbol{m}_\mathrm{c} \cdot \boldsymbol{q} \\ d_\mathrm{d} &= \boldsymbol{m}_\mathrm{d} \cdot \boldsymbol{q} \\ d_\mathrm{w} &= \boldsymbol{m}_\mathrm{w} \cdot \boldsymbol{q} \end{aligned} \tag{10}$$

Here, c is the modeled concentration (in $\mathrm{kg\,m^{-3}}$), d_d the dry deposition rate and d_w the wet deposition rate (both in $\mathrm{kg\,m^{-2}\,s^{-1}}$). In this specific case with only a single scalar receptor, the source–receptor matrix degenerates to a vector of the SRR values, one for each of the three types of receptor ($\boldsymbol{m}_\mathrm{c}$ for concentration in units of seconds, $\boldsymbol{m}_\mathrm{d}$ for dry deposition and $\boldsymbol{m}_\mathrm{w}$ for wet deposition, both in units of meters). In order to obtain the concentration or the deposition rates, these vectors need to be multiplied with the vector of emissions $\boldsymbol{q}$ ($\mathrm{kg\,m^{-3}\,s^{-1}}$). If the total deposition is desired, the deposition rates d_d and d_w can be multiplied with the receptor time interval ΔT_r, in our case 86 400 s (24 h). Note that this is the period during which particles are released according to the specification of the `RELEASES` file. The emission fluxes must be volume averages over the output grid cells specified in the `OUTGRID` file, typically surface emission fluxes (in $\mathrm{kg\,m^{-2}\,s^{-1}}$) divided by the height of the lowermost model layer.

2.6 Sensitivity to initial conditions

Backward simulations with FLEXPART in the context of inverse modeling problems typically track particles for several days up to a few weeks. This is sufficient to estimate concentrations at the receptor only for species with atmospheric lifetimes shorter than this period. Many important species (e.g., greenhouse gases such as methane) have considerably longer lifetimes. For such long-lived species, most of the atmospheric concentration variability is still caused by emission and loss processes occurring within the last few days before a measurement because the impact of processes occurring at earlier times is smoothed out by atmospheric mixing. This leads to a relatively smooth "background" (in time series analyses sometimes also called a baseline) that is often a dominant fraction of the total concentration but that does not vary much with time, with short-term fluctuations on top of it. The signal of the regional emissions around the measurement site is mostly contained in the short-term concentration fluctuations but in order to use it in inverse modeling, the background still needs to be accounted for, as otherwise no direct comparison to measurements is possible.

One simple method is to estimate the background from the measurements as, e.g., in Stohl et al. (2009). A better approach is to use a concentration field taken from a long-term forward simulation with an Eulerian model or with FLEXPART itself, especially if nudged to observations (Groot Zwaaftink et al., 2018), as an initial condition for the backward simulation. This field needs to be interfaced with the FLEXPART backward simulation by calculating the receptor sensitivity to the initial conditions (see Eqs. 2–6 in Seibert and Frank, 2004). For instance, for a 10 d backward simulation, the concentration field needs to be sampled at those points in time and space when and where each particle trajectory terminates 10 d back in time. Furthermore, it is necessary to quantify the effects of deposition or chemical loss during the backward simulation on this background (the factor $p(0)$ in Seibert and Frank, 2004). For example, chemical reactions with hydroxyl radicals will reduce initial concentrations of methane en route to the receptor, even though this is not much during a 10 d period.

Since version 8.2, FLEXPART has provided an option to quantify the influence of initial conditions on the receptor in backward simulations, which is activated with the switch `LINIT_COND` in file `COMMAND`. Then, gridded fields containing the sensitivities to background mixing ratios (or concentrations, depending on user settings for the switch `LINIT_COND` in file `COMMAND`) are produced and stored in the output files `grid_initial_nnn` (`nnn` stands for the

species number) on the same 3-D grid as the regular output, defined in the files OUTGRID and OUTGRID_NEST. In this case, a concentration would be calculated as

$$c = \boldsymbol{m}_{\mathrm{i}} \cdot \boldsymbol{c}_{\mathrm{b}} + \boldsymbol{m}_{\mathrm{c}} \cdot \boldsymbol{q}, \tag{11}$$

where $\boldsymbol{m}_{\mathrm{i}}$ denotes the sensitivity to the initial condition and $\boldsymbol{c}_{\mathrm{b}}$ the background concentration when and where particles are terminated.

Figure 4 shows an example of the use of the sensitivities of receptor mixing ratios (here, of methane) to both surface emissions and initial conditions. The panel (b) shows the sensitivity to surface emissions on one particular day, and the panels (c) and (d) show the sensitivity to initial conditions below and above 3000 m for the same day. Both results are from an 8 d backward simulation from one receptor site in Sweden. It can be seen that the sensitivity to emissions is highest close to the station, but there is also substantial sensitivity to emission uptake over large parts of central and eastern Europe. The particles terminated 8 d before arrival at the receptor in a roughly croissant-shaped area covering large parts of Europe and the North Atlantic, as indicated by the sensitivity to initial conditions. Most of the sensitivity is located below 3000 m but there is also some influence from higher levels. Notice that only two layers are shown in Fig. 4, whereas the real model output has much higher vertical resolution.

The sensitivity to initial conditions was interfaced with a domain-filling methane forward simulation as described in Groot Zwaaftink et al. (2018) (not shown), while the emission sensitivity was interfaced with an emission inventory for methane (not shown), as given by Eq. (11). This was done for daily simulations throughout 1 month, thus generating a time series of background mixing ratios (from the first term in Eq. 11 only) and total mixing ratios (Fig. 4a). The latter include the contributions from emissions during the 8 d backward simulation. It can be seen that the methane background advected from 8 d back varies relatively little between about 1910 and 1940 ppbv, while the emission contributions vary from 0 (on 29 October) to about 200 ppbv (on 19 October, the date for which the sensitivity plots are shown).

In practical applications for inverse modeling, source–receptor sensitivities are often only needed at the surface (as most emissions occur there), while sensitivities to the background are needed in 3-D. By setting the option SURF_ONLY to 1 in the COMMAND file, the regular output files grid_time_date_nnn containing the source–receptor sensitivities will include only the first vertical level as defined in the file OUTGRID, while the full vertical resolution is retained in grid_initial_nnn files containing the sensitivities to the initial conditions. Since the data amounts stored in the grid_time_date_nnn files can be much larger than in the grid_initial_nnn files, this is a highly efficient way to save storage space. This setup also interfaces directly with the inverse modeling package FLEXINVERT (Thompson and Stohl, 2014). An application can be found in Thompson et al. (2017) wherein initial conditions were taken from a gridded observation product. A further output option, which was also introduced for practical considerations of inverse modeling, is the LINVERSIONOUT option in the file COMMAND. If LINVERSIONOUT is set to 1, then the grid_time_date_nnn and grid_initial_nnn files are written per release with a time dimension of footprints instead of the default per footprint with a time dimension of releases. Since inverse modeling assimilates atmospheric observations and each observation is represented by a single release, it is computationally more efficient to read in the grid files separated by release. This output format also interfaces directly with FLEXINVERT.

2.7 Chemical reactions with the hydroxyl radical (OH)

The hydroxyl (OH) radical reacts with many gases and is the main cleansing agent in the atmosphere. While it is involved in highly nonlinear atmospheric chemistry, for many substances (e.g., methane) a simplified linear treatment of loss by OH is possible using prescribed OH fields. For this, monthly averaged $3° \times 5°$ resolution OH fields for 17 atmospheric layers are used in FLEXPART. The fields were obtained from simulations with the GEOS-Chem model (Bey et al., 2001) and are read from the file OH_variables.bin by the subroutine readOHfield.f90.

Tracer mass is lost by reaction with OH if a positive value for the OH reaction rate is given in the file SPECIES_nnn. In FLEXPART v10.4, the OH reaction scheme was modified to account for (i) hourly variations in OH and (ii) the temperature dependence of the OH reaction rate (Thompson et al., 2015). This makes the chemical loss calculations more accurate, especially for substances with shorter lifetimes (of the order of weeks to months), for example ethane. Hourly OH fields are calculated from the stored monthly fields by correcting them with the photolysis rate of ozone calculated with a simple parameterization for cloud-free conditions based on the solar zenith angle (gethourlyOH.f90):

$$\mathrm{OH} = \frac{j}{j^*}\,\mathrm{OH}^*, \tag{12}$$

where j represents the hourly photolysis rates calculated for all 3-D locations in the field, while j^* represents the corresponding monthly mean rates, precalculated and stored in file OH_variables.bin together with the monthly mean fields OH^* (see Sect. 5.1.8). The motivation for this is that OH production closely follows the production of $\mathrm{O}(^1\mathrm{D})$ by the photolysis of ozone, allowing for this simple parameterization of OH variability. At any time, two hourly OH fields are in memory and are interpolated to the current time step. Figure 5 shows the annual and daily variation of OH for two locations as obtained with this simple parameterization.

The OH reaction rate κ (s^{-1}) is calculated in ohreaction.f90 using the temperature-dependent

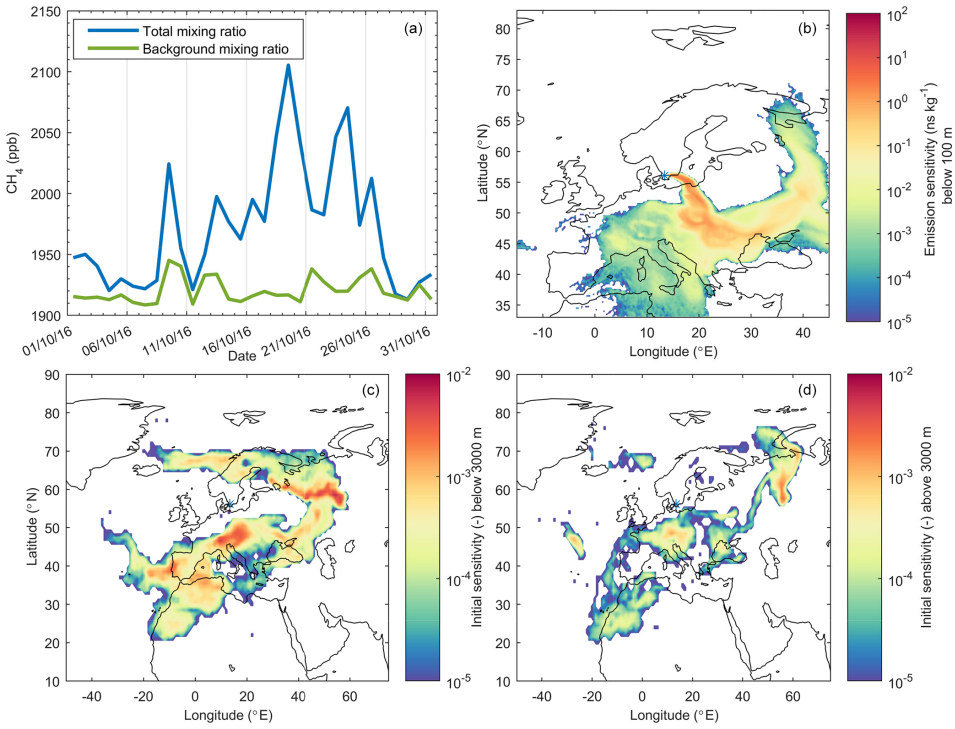

Figure 4. Example of FLEXPART 8 d backward runs for methane from a site in southern Sweden (Hyltemossa) demonstrating the combined use of sensitivities to emissions and initial conditions. **(a)** Time series of methane background mixing ratios and total mixing ratios in October 2016. **(b)** Sensitivity of the methane mixing ratio at Hyltemossa on 19 October 2016 to methane emissions at the surface. **(c)** Sensitivity of the methane mixing ratio at Hyltemossa on 19 October 2016 to methane initial conditions below 3000 m. **(d)** Sensitivity of the methane mixing ratio at Hyltemossa on 19 October 2016 to methane initial conditions above 3000 m. Blue asterisks on the maps mark the receptor location.

formulation

$$\kappa = C\,T^{N} e^{-D/T}\,[\mathrm{OH}], \tag{13}$$

where C, N and D are species-specific constants (assigned in the `SPECIES_nnn` files), T is the absolute temperature, and [OH] the OH concentration (Atkinson, 1997). As the OH concentration in file `OH_variables.bin` is given in units of molecules per cubic centimeter, the unit of C needs to be in cubic centimeters per molecule per second ($\mathrm{cm}^3\,\mathrm{molecule}^{-1}\,\mathrm{s}^{-1}$). The mass m of a given species after reaction with OH is determined as

$$m(t + \Delta t') = m(t)\, e^{-\kappa \Delta t'}, \tag{14}$$

where $\Delta t'$ is the reaction time step (given by `lsynctime`).

Backwards compatibility with the former temperature-independent specification of the OH reaction (version 9 and before) can be achieved by setting the constant N in the `SPECIES_nnn` file to zero. The constants C and D can be derived from the former parameters as follows:

$$C = \kappa_{\mathrm{r}}\, e^{D/T_{\mathrm{r}}} \tag{15}$$

and

$$D = A/\mathrm{R}, \tag{16}$$

where A is the activation energy, R is the gas constant and κ_{r} is the former OH reaction rate (referring to $T_{\mathrm{r}} = 298$ K), which were specified in the `SPECIES_nnn` file for earlier versions.

OH fields other than those provided with the model code have been tested in FLEXPART. These fields may have higher spatial and temporal resolution (e.g., Fang et al., 2016), which is important for chemical species with short lifetimes. Users are required to modify `readOHfield.f90` and `gethourlyOH.f90` to read in other OH fields and be aware that expressions of the OH reaction rate or reaction with OH might differ from those in the above equations. If this is the case users need to modify `ohreaction.f90`, too.

2.8 Dust mobilization scheme

Desert dust is a key natural aerosol with relevance for both climate and air quality. FLEXPART has been used earlier with preprocessors to initialize dust amounts from wind speed and surface properties following Tegen and Fung (1994) (Sodemann et al., 2015). Now a dust mobilization routine has been included as a preprocessing tool in FLEXPART v10.4. The scheme, called FLEXDUST, was devel-

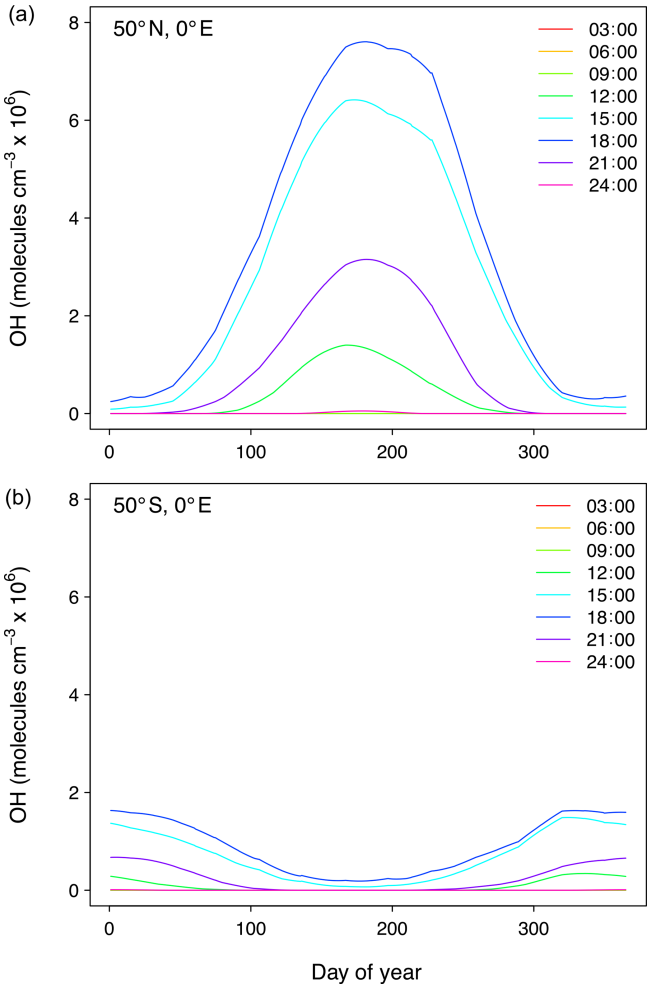

Figure 5. Annual and daily OH concentration variation as obtained with the simple parameterization based on photolysis rates of ozone for two locations, one in the Northern Hemisphere **(a)** and one in the Southern Hemisphere **(b)**. Line labels correspond to the time of day.

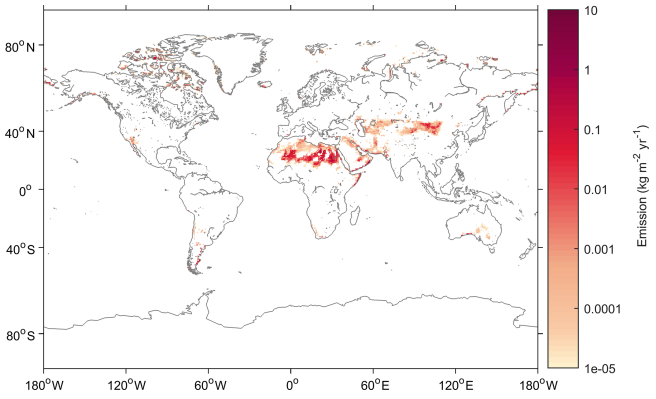

Figure 6. Average annual dust emission for the period 1990–2012 estimated with FLEXDUST driven with ERA-Interim meteorology.

oped to simulate mineral dust transport with FLEXPART in forward or backward simulations (Groot Zwaaftink et al., 2016). This module runs independently from FLEXPART and produces gridded output of mineral dust emissions as well as input files (`RELEASES`) that can be used for FLEXPART simulations of atmospheric transport. It can thus be considered a preprocessing (for forward simulations) or post-processing tool (for backward simulations) for FLEXPART v10.4.

In FLEXDUST, emission rates are estimated according to the emission scheme proposed by Marticorena and Bergametti (1995). We thereby assume that sandblasting occurs in the case that sand is present and a minimum threshold based on the size-dependent threshold friction velocity following Shao and Lu (2000) can be applied. The following are used as input for the model: ECMWF operational analysis or ERA-Interim reanalysis data, Global Land Cover by National Mapping Organizations version 2 (Tateishi et al., 2014), and and sand and clay fractions from the Global Soil Data Task (2014). Erodibility is enhanced in topographic depressions, and dust emissions are modified by soil moisture and snow cover. The module includes high-latitude dust sources in the Northern Hemisphere. These sources are rarely included in global dust models, even though they appear important for the climate system and substantially contribute to dust in the Arctic (Bullard et al., 2016; Groot Zwaaftink et al., 2016). Icelandic deserts in particular are known to be highly active, and a high-resolution surface type map for Iceland can therefore be included in FLEXDUST simulations (Arnalds et al., 2016; Groot Zwaaftink et al., 2017). Like in FLEXPART, nested meteorological fields can be used for specific regions of interest. The size distribution of emitted dust follows Kok (2011), is independent of friction velocity and is by default represented by 10 size bins. This can be changed depending on known properties or assumptions of dust sources. The dust particles are assumed to be spherical in FLEXPART. An example of annual mean dust emissions from 1990 to 2012 calculated with FLEXDUST driven with ERA-Interim meteorology is shown in Fig. 6. Further details on FLEXDUST, including model evaluation, are given by Groot Zwaaftink et al. (2016). The source code is available from the git repository: https://git.nilu.no/christine/flexdust.git (last access: 30 October 2019).

3 Parallelization

In a Lagrangian model like FLEXPART, particles move totally independently of each other. This facilitates efficient parallelization of the code. The most simple and often most effective way is running several instances of the model in parallel. For example, if the model is to be run backwards (for 10 d, for example) at regular intervals from a measurement site for a year, one could run the model separately, in par-

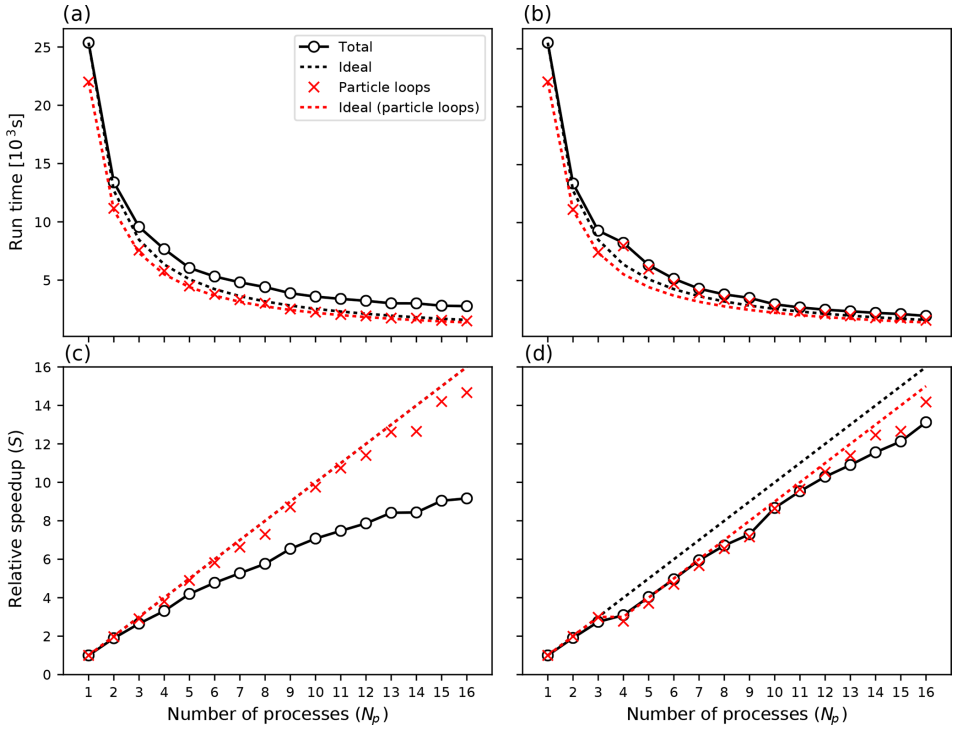

Figure 7. Computational time **(a, b)** and speedup **(c, d)** for up to 16 processes on a single node. In panels **(a, c)**, all processes read meteorological input data, whereas in panels **(b, d)**, a dedicated process reads and distributes input data for $N_\mathrm{p} \geq 4$.

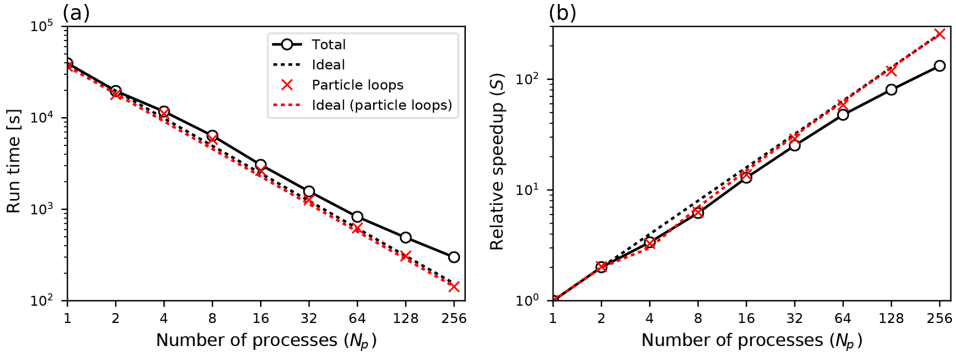

Figure 8. Computational time **(a)** and speedup **(b)** for up to 256 processes on 16 nodes. Logarithmic scaling along both axes. For $n \geq 4$ a dedicated process reads and distributes input data.

allel, for monthly subperiods. The total computation time of the 12 monthly processes together is nearly the same as if the model is run as one process for the whole year. Some overhead in processing input data occurs because, in the above example, 10 extra days of data per process are needed to calculate trajectories 10 d back into the preceding month. One disadvantage of that approach is that the memory needed for holding the meteorological input data and the model output fields is multiplied. However, this overhead is often small; thus, this approach has been used very often by FLEXPART users in the past.

Even if a task cannot easily be decomposed into runs for different periods or sources, trivial parallelization is still possible if a large number of particles is desired, for example in a domain-filling simulation for which tens of millions of particles may be used. The strategy in this case would be to assign a fraction of the particles to each run. Note that different random seeds should be used for each run, which requires a manual change and recompilation of the code.

As a user-friendly alternative, FLEXPART v10.4 has been parallelized using standard parallelization libraries. Common parallelization libraries are Open Multi-Processing (OpenMP; http://www.openmp.org/, last access: 30 Octo-

ber 2019), which is designed for multicore processors with shared memory, and Message Passing Interface (MPI, 2015) for distributed memory environments. Examples of other Lagrangian particle models that have been parallelized are NAME (Jones et al., 2007) and FLEXPART–WRF (Brioude et al., 2013), which use a hybrid approach (OpenMP+MPI). For FLEXPART v10.3 we decided to use a pure MPI approach for the following reasons.

- It is simpler to program than a hybrid model and more flexible than a pure OpenMP model.
- While OpenMP in principle may be more effective in a shared memory environment, MPI can often perform equally well or better provided there is not excessive communication between the processes.
- MPI offers good scalability and potentially low overhead when running with many processes.

3.1 Implementation

The FLEXPART code contains several computational loops over all the particles in the simulation, which is where most of the computational time is spent for simulations with many particles. The basic concept behind our parallel code closely resembles the "trivial parallelization" concept described above. When launched with a number of processes, N_p, each process will separately calculate how many particles to release per location, attempting to achieve an approximately even distribution of particles among the processes while keeping the total number of particles the same as for a simulation with the serial version. Each running process will generate an independent series of random numbers and separately calculate trajectories and output data for its set of particles. Explicit communications between processes are only used when the output fields are combined at the master process (MPI rank 0) using `MPI_Reduce` operations, before writing the output. Also, in the case in which the output of all individual particle properties is desired (option `IPOUT1 = 1` or `2` in file `COMMAND`), we let each process append its data to the same file. We thus avoid the costly operation of transferring particle properties between processes. The performance of the implementation is discussed in Sect. 3.2 (see Fig. 7).

Some parts of the code are not simply loops over particles, most notably the routines for reading and transforming the input meteorological data. It follows that the performance gain of using parallel FLEXPART in general is better for simulations with a larger number of particles. We have, however, implemented a feature whereby instead of having each MPI process read and process the same input data, one dedicated MPI process is set aside for this purpose. When the simulation time t lies in the interval between wind field time T_i and T_{i+1}, all other processes calculate particle trajectories, while this dedicated process ingests input fields from time T_{i+2}. At simulation time $t = T_{i+1}$ the dedicated "reader process" will distribute the newest data to the other processes and immediately start reading fields for time T_{i+3}, while the other processes continue doing trajectory calculations. A hard-coded integer (`read_grp_min` in file `mpi_mod.f90`) is used to set the minimum number of total MPI processes for which this separate process will be reserved for reading input data. For the examples shown in Sect. 3.2 a value of 4 was used (Figs. 7 and 8).

3.2 Performance aspects

To assess the performance of the parallel code we performed three scaling experiments of various size on different computational platforms.

3.2.1 40 million particles, single 16-core node

In the following we present the results from running the code on a machine equipped with an Opteron 6174 processor with 16 cores. Compilation was done using gfortran version 4.9.1 and OpenMPI version 1.8.3. For the experiment, 40 million particles were released and propagated 48 h forward in time. We ran with this setup with an increasing number of processes, from 1 to 16. All time measurements in the code were made with the `MPI_wtime()` subroutine.

For the first experiment, every process separately processed the meteorological input data. Figure 7a and c show the CPU time T_n used in the case of n processes and the relative speedup factor $S(n) = T_1/T_n$. Time and speedup shown for "particle loops" includes the three most computationally demanding particle loops (integration of the Langevin equation, wet deposition and concentration calculations), but, in addition, FLEXPART contains a few smaller loops over particles that exhibit similar performance improvements. We see that for 40 million particles, the loops over particles take the largest share, at least 87 % of the total time when run with one process. Close-to-perfect speedup is expected and observed for these loops (compare results for "particle loops" and "ideal (particle loops)" in Fig. 7a and c). The major bottleneck for overall performance in this case is that each process reads the same input files from disk, thus forcing the others to wait. This bottleneck causes the speedup to deviate substantially from the ideal situation when more than a few processes are used (compare results for "total" and "ideal" in Fig. 7a and c).

Next we repeated the experiment above but set aside a dedicated process for reading the meteorological data whenever $n \geq 4$. The results are shown in Fig. 7b and d. Numerical values for the speedup factors for selected numbers of processes are given in Table 3. We observed that with $n \geq 7$ there was consistently a benefit to setting aside the dedicated reader process, whereas for $n < 7$ it was more effective to have all processes read data and thus an extra process available for doing the trajectory calculations. These results will of course

Table 3. Computational speedup S for up to 16 processes (single-node experiment) for the two different MPI modes, with 40 million released particles.

Number of processes	1	2	4	8	16
All processes read	1	1.89	3.30	5.76	9.16
Dedicated read process	1	1.91	3.09	6.72	13.10

vary with the resolution of the input data, the number of particles and the system on which the program is run.

3.2.2 500 million particles, multiple 16-core nodes

We performed a larger-scale experiment at the Abel computer cluster[1] using up to 256 cores on 16 nodes with Intel Xeon E5-2670 CPUs. For each node, up to 16 cores were used, and then the number of nodes was determined by the total number of processes launched. The FLEXPART setup was similar to the previous single-node experiment, but we increased the number of particles to 500 million and reduced the simulated time to 12 h. Compilation was done with Intel Fortran v16.0.1 and OpenMPI v1.10.2.

Run time and speedup factors are shown in Table 4 and Fig. 8. As before we see essentially perfect speedup of the computationally intensive parts (the particle loops), which is expected. Table 4 also gives the parallel efficiency, which is seen decreasing for larger N_p. This is partly due to the increased cost of MPI communications and also because the nonparallel parts of the code have relatively higher impact. With 256 processes there are only about 2 million particles per process and the CPU time is not as clearly dominated by the particle loops as when 500 million particles all run in one process. In addition, the initialization of the code (allocation of arrays, reading configuration files) takes around 20 s for this run, which is significant given a total run time of 299 s. Thus, parallel efficiency would increase for longer simulation times and/or for simulations with more particles per process, i.e., realistic cases that are more likely to be run with such a large number of processes.

3.2.3 900 000 particles, laptop and single 16-core node

Finally, we examined a small-scale experiment in which we released 900 000 particles and simulated 15 d of transport. The performance was tested on two systems; a ThinkPad P52s laptop (Intel i7-8550U CPU with four cores; results in Table 5) and a machine equipped with an AMD Opteron 6386 SE processor (16 cores; results in Table 6). With this relatively lower number of particles it is not surprising to see that the parallel efficiency is lower than in the preceding examples. Still, we see that a speedup of 2.38 on a 4-core laptop and 5.25 on a 16-core machine is attainable. We also note that for practical applications, users would likely use the serial version for applications with so few particles and, if there are many such runs to be done, use trivial parallelization by submitting many separate serial runs in parallel. The parallelization feature is most useful for cases with a very large number of particles that cannot so easily be split in many separate runs, such as domain-filling simulations.

[1] Owned by the University of Oslo and Uninett/Sigma2, operated by the Department for Research Computing at USIT, the University of Oslo IT department; https://www.uio.no/english/services/it/research/hpc/abel/ (last access: 30 October 2019).

3.3 Validation

In order to ensure that the parallel version produces results with the same accuracy as the serial version, we have performed a set of tests and validation experiments. A direct comparison between the versions can only be performed in statistical terms because FLEXPART uses Gaussian-distributed random numbers for calculating the turbulent velocities of the particles. For the parallel version we let each process independently calculate a set of random numbers, which leads to small numeric differences (arising from the random "noise") between the versions.

To confirm that the only source of differences between the serial and parallel code is in the random number generation, we first observe that when the parallel executable is run using only one process, it produces results identical to the serial version. This is as expected, as the first MPI process (rank 0) always uses the same random number seeds as the serial version.

Next, we have done tests in which all random numbers are set to zero in both codes, corresponding to switching off the turbulent displacements, and we run the parallel version using multiple processes. The outputs from the serial and parallel versions of the code when run this way are identical except for small differences due to round-off errors (e.g., in concentration calculations – these round-off errors are typically larger in the serial version due to the larger number of particles).

4 Installation, compilation and execution

FLEXPART is usually used in a Linux environment, which we also assume for the following instructions. However, the model has also been implemented successfully under MacOS and MS Windows. The default Fortran compiler for FLEXPART v10.4 is gfortran, but ifort, Absoft and PGI compilers have been used as well.

4.1 Required libraries and FLEXPART download

As the meteorological data from numerical weather prediction models are usually distributed in GRIB format, a library for reading GRIB data is required. It is recommended to use `ecCodes` (https://software.ecmwf.int/wiki/display/ECC, last access: 30 October 2019), the primary GRIB

Table 4. Run time and speedup for the multi-node experiment with 500 million particles. Up to 16 nodes in the Abel cluster (University of Oslo).

Number of processes n	1	2	4	8	16	32	64	128	256
Total run time (s)	39 536	19 681	14 123	6380	3061	1568	828	491	299
Speedup factor S	1	2.01*	3.37	6.20	12.92	25.22	47.76	80.53	132.12
Parallel efficiency (S/n)	1	1.004	0.843	0.775	0.807	0.788	0.746	0.629	0.516

* Superlinear speedup (efficiency greater than 100 %) as seen here is usually attributed to memory and/or cache effects.

Table 5. Run time and speedup using up to four cores on a ThinkPad P52s laptop (900 000 particles).

Number of processes n	1	2	3	4
Total run time (s)	3504	2035	1790	1470
Speedup factor S	1	1.72	1.97	2.38
Parallel efficiency (S/n)	1	0.86	0.66	0.775

encoding–decoding package used at the ECMWF (recent enough versions of its predecessor `grib_api`, no longer supported after 2018, can also be used). Data in GRIB-2 format can be compressed. If this is the case for the input data, the `jasper` library is needed[2]. If it is desired to produce FLEXPART output in the NetCDF format, the NetCDF Fortran Library (https://www.unidata.ucar.edu/software/netcdf/, last access: 30 October 2019) is also required.

In order to obtain the FLEXPART source code, download the appropriate v10.4 tarball from the FLEXPART website[3] and unpack it.

```
tar -xvf flexpart10.4.tar
```

To obtain the latest available model version, clone the FLEXPART git repository from the FLEXPART community website.

```
git clone https://www.flexpart.eu/
gitmob/flexpart
```

This repository mirrors https://transport.nilu.no (https://git.nilu.no/flexpart/flexpart/-/releases, last access: 18 November 2019). Additional mirrors exist, e.g., at GitHub (https://github.com/flexpart/flexpart, last access: 30 October 2019) and BitBucket (https://bitbucket.org/flexpart/flexpart, last access: 30 October 2019). See the "Code and data availability" section for additional information. After unpacking the tarball or cloning the repository, a local directory structure as shown in Table 1 is created.

[2]The `jasper` package is available in Linux distributions; https://github.com/mdadams/jasper (last access: 30 October 2019).

[3]The website, https://flexpart.eu/ (last access: 30 October 2019), provides additional information that can be used to supplement these instructions.

The directory `src` contains the code and a makefile. The makefile needs to be adapted to the compiler and libraries present on the local system. Appendix A4 describes these steps in detail, including manual installation of the libraries. This was tested for Ubuntu 16.04.3 as well as 18.04.3 LTS Linux and MacOS (OS versions 14.5.0 and 18.6.0). Both a serial and a parallel executable can be built from the FLEXPART v10.4 source files.

4.2 Compiling and running the serial version

After correctly setting the library paths in the makefile, the command `make` produces the executable called `FLEXPART`. It can be executed from the command line by `./FLEXPART` and then expects a file `pathnames` to exist in the current working directory. This file contains information on where input data are located and where output data shall be produced (see Sect. 5). Note that `pathnames` is expected in the directory from which FLEXPART is started, which can be different from where the executable file is located. A different name of a `pathnames` file can be also given as an argument. FLEXPART can thus be invoked according to the following generic syntax:

```
path_to_flexpart/flexpart_executable
path_to_pathnames/pathnames_file.
```

Using an optional argument, `-v` (verbose mode), will display additional information during the run. Even more information, including clock time between different program units, will be printed with `-v2`. Invoking FLEXPART with the flags `-i` and `-i2` (info mode) will provide detailed run-specific information while reading input files. However, in this mode FLEXPART then stops before particle trajectories are calculated.

4.3 Compiling and running the parallel version

Most subroutines calling MPI functions are in a single module named `mpi_mod.f90`. Other FLEXPART source files that depend on this module are given the `_mpi.f90` suffix to distinguish them from the serial version. During compilation the makefile selects the source files automatically depending on whether the parallel or serial version is built.

In order to compile and run the parallel version, an MPI library must be installed, either as a package from

Table 6. Run time and speedup using up to 16 cores on a machine equipped with an AMD Opteron 6386 SE processor (900 000 particles).

Number of processes n	1	2	4	6	8	10	12	16
Total run time (s)	3788	2337	1376	976	840	765	747	717
Speedup factor S	1	1.62	2.75	3.88	4.51	4.95	5.07	5.29
Parallel efficiency (S/n)	1	0.81	0.69	0.65	0.56	0.50	0.42	0.33

the distribution or built from source code. Both OpenMPI (http://www.open-mpi.org/software/ompi/v1.8/, last access: 30 October 2019) and mpich2 (http://www.mpich.org/downloads/, last access: 30 October 2019) work, but testing on some systems indicates slightly better performance with OpenMPI. As for the other libraries, the MPI library names and paths need to be adapted in the makefile. The `MPIF90` variable sets the Fortran compiler wrapper (usually `mpifort` or `mpif90`; in the case of coexisting OpenMPI and mpich installations, wrappers called `mpif90.openmpi` or `mpif90.mpich` may be defined). Compilation of the parallel version should then be done by the following.

```
make mpi
```

This will produce an executable file `FLEXPART_MPI`. If executed, this will run on a single processor and should produce results identical to the serial version. To activate the parallel features, the executable must be run through an MPI launcher (here it is important to use the launcher corresponding to the MPI library that was used for the compilation).

```
mpirun -n <number> FLEXPART_MPI
```

In this example, `<number>` specifies the number of processes one wishes to launch. For some installations, `mpirun` is called `mpiexec`, or, in the case of coexisting OpenMPI and mpich2 installations, `mpiexec.openmpi` or `mpiexec.mpich2`, respectively. Many command-line options exist for `mpirun` that can be helpful for improving performance, e.g., processor binding. For a list of these options, see `mpirun -help`. In practice, the optimal number of cores for a given simulation will depend on the size of the problem and the hardware availability among other factors.

5 FLEXPART input

In this section, we describe the different FLEXPART input files and, where appropriate, changes that have occurred since the last publication (Stohl et al., 2005). FLEXPART needs the following three types of input files.

1. The text file `pathnames` is located by default in the directory where FLEXPART is executed. It must contain at least four lines: first, the path to the directories where run-defining input files are located (the so-called `options` directory); second, the path where output files are created; third, the path to the meteorological input GRIB files; and, fourth, the path to the so-called `AVAILABLE` file (see point 3). The last two lines can be repeated if nested input data shall be used. For each nesting level, one line for the GRIB data directory and one for the corresponding `AVAILABLE` file are needed.

2. The files containing the run-defining settings are located in a subdirectory (given in line 1 of `pathnames`) by default called `options` (see Table 1). The settings, which control FLEXPART's physics and program flow, are stored in different text files listed in Table 7 and described in Sect. 5.1. In addition, the `options` directory contains data files that are not usually changed by the user.

3. The meteorological input data, one file for each input time, are stored in GRIB format in a common directory (specified in line 3 of `pathnames`). To enable FLEXPART to find these files, a file usually named `AVAILABLE` (given in line 4 of `pathnames`) contains a list of all available meteorological input files and their corresponding time stamps. Additional files containing nested input data may also be provided. In this case, a separate file containing the input file names (e.g., named `AVAILABLE_NESTED`) must be given. Date and time entries in the `AVAILABLE*` files for mother and nested fields must be identical. Details on the meteorological input data are given in Sect. 5.2.

5.1 Run-defining settings: the `options` directory

Here, we give an overview of the information provided in the run-defining FLEXPART user input files listed in Table 7. In previous versions of FLEXPART, these files were formatted text files (coming alternatively in a long and a short format). For backward compatibility, these plain text formats are still supported. However, FLEXPART v10.4 also allows for the use of namelists, a standard Fortran feature whereby values are provided in a list with elements of the form `name=value`. When FLEXPART is started, it tries to open the files as namelists, and if this is not working, it expects the files to be in one of the two old plain text formats. We encourage users to update their input files to namelists for two reasons. Firstly, FLEXPART now has default user options for all input settings so that users only need to set those options that they want to deviate from the defaults. Secondly,

Table 7. Alphabetical list of the run-defining input files (upper part) and static input files (lower parts), usually contained in a directory called `options`. Processing of files marked with * depends on the run specifications. The other files are always read in.

File name	Content
AGECLASSES*	Age class definitions
COMMAND	Main control parameters
OUTGRID	Output grid definition
OUTGRID_NEST*	Nested output grid definition
RECEPTORS*	Receptor locations for receptor kernel output
RELEASES	Specification of the sources (forward run) or receptors (backward run)
SPECIES/	Directory containing files with definitions of physical and chemical parameters of species referenced in `RELEASES`
IGBP_int1.dat	Land cover input data
surfdata.t	Roughness length, leaf area index for different land cover types
surfdepo.t	Seasonal surface resistances for different land cover types
OH_variables.bin*	OH field

namelists make it easier to add new user options, which may be required in future versions of FLEXPART. Thus, plain text input files may not be supported in future versions of FLEXPART. Examples for all formats of the user input files are contained in the FLEXPART distribution.

To convert user input files of any format to namelist format, the switch `nmlout=.TRUE.` (in file `com_mod.f90`) must be set before compilation. Then, run-defining user input files are written out in namelist format in the `output` directory, with the appendix `.namelist` added to the input file name (e.g., `COMMAND.namelist`). This feature also improves the traceability of FLEXPART model results and makes simulations easily reproducible by documenting the settings used for the model run.

In the following, we provide reference tables of the run-defining user input files including default settings (in bold) when using the namelist format. Notice that the default values are appropriate for regional-scale simulations, but simulations on smaller scales or with higher accuracy may need adjustments (in particular, shorter time steps and the use of the new CBL scheme).

5.1.1 File `COMMAND`

The `COMMAND` file contains the user settings controlling the simulation and the behavior of the run. The default `COMMAND` file contains a namelist `&COMMAND`, for which Table 8 provides a complete listing of all settings with their meaning and preset default values. It is important that users of previous FLEXPART versions who choose to use plain text input files update their `COMMAND` file, since new parameters have been added. However, the `cblflag` (and any option added afterwards) must be provided in namelist format in any case.

5.1.2 File `RELEASES`

The `RELEASES` file contains information related to when and where the particles are introduced in the simulation and other properties of the release points (e.g., the chemical species simulated). It consists of a namelist `&RELEASES_CTRL` that specifies header information. The header gives the total number of different species (i.e., different substances) to be released, followed by a corresponding list of the FLEXPART species numbers *nnn*, and `SPECIES_nnn` files define the species' physical properties (see Sect. 5.1.3). Following the header, there is an arbitrary number of namelists `&RELEASE` defining each release. For each such release, the following is given: the starting and ending time, the location and extension, the masses released (one value for each released species), and the number of particles to be released, as well as a comment string. The content of the `RELEASES` file is summarized in Table 9.

5.1.3 `SPECIES` files

The subdirectory `options/SPECIES/` needs to contain one or more files named `SPECIES_nnn`. For each species *nnn* listed in the header section of the `RELEASES` file, such a `SPECIES_nnn` file must exist. The parameters in the `SPECIES_nnn` file, contained in the namelist `&SPECIES_PARAMS`, set the species name and define the physicochemical properties of the species; they are described in Table 10. These are important for simulating radioactive or chemical decay, wet deposition (scavenging) for gases and aerosols, dry deposition for gases and aerosols, particle settling, and chemical reaction with the OH radical. Some parameters are only necessary for gas tracers and some are only necessary for aerosol tracers; thus, a namelist does not need to contain all parameters for both gases and particles. Optionally, since FLEXPART version 6.0, information about temporal emission variations can be added at the end of the file.

Table 8. Contents of the user input file COMMAND. Variable names with their meaning and all possible values are listed. Where appropriate, default values are given in bold. Note that not all input parameter combinations are allowed.

	Variable name	Description	Value (default)
1	LDIRECT	Simulation direction in time	**1 (forward)** or −1 (backward)
2	IBDATE	Start date of the simulation	YYYYMMDD: YYYY=year, MM=month, DD=day
3	IBTIME	Start time of the simulation	HHMISS: HH hours, MI=minutes, SS=seconds. UTC zone.
4	IEDATE	End date of the simulation	Same format as IBDATE
5	IETIME	End time of the simulation	Same format as IBTIME
6	LOUTSTEP	Interval of model output	Average concentrations are calculated every LOUTSTEP (10 800 s)
7	LOUTAVER	Concentration averaging interval, instantaneous for value of zero	**(10 800 s)**
8	LOUTSAMPLE	Numerical sampling rate of output, higher statistical accuracy with shorter intervals	**(900 s)**
9	ITSPLIT	Time constant for particle splitting (particles are split in two after a given time)	**(999 999 999 s)**
10	LSYNCTIME	All processes are synchronized to this time interval; it has to divide all values above	**(900 s)**
11	CTL	Factor by which particle transport time step in the ABL must be smaller than the Lagrangian timescale t_l; resulting time steps can be shorter than LSYNCTIME; LSYNCTIME is used if CTL < 0	> 1 for time steps shorter than t_l; if CTL < 0, a purely random walk simulation is done **(–5.0)**
12	IFINE	Additional reduction factor for time step used for vertical transport only considered if CTL > 1	Positive integer **(4)**
13	IOUT	Switch determining the output type	**(1)** mass concentration (residence time backwards), 2 volume mixing ratio, 3 both 1 and 2, 4 plume trajectories, 5 both 1 and 4. Add 8 for NetCDF output
14	IPOUT	Switch for particle position output	**(0)** no particle output, 1 particle output every output interval, 2 only at the end of the simulation (useful, e.g., for warm start)
15	LSUBGRID	Increase in ABL heights due to subgrid-scale orographic variations	**(0)=off**, 1=on
16	LCONVECTION	Switch for convection parameterization	0=off, **(1)=on**
17	LAGESPECTRA	Switch for calculation of age spectra (needs file AGECLASSES)	**(0)=off**, 1=on
18	IPIN	Warm start simulation, restarted from a particle dump (needs partposit_end file from previous simulation)	**(0)=no**, 1=yes
19	IOER	Separate output fields for each location in the RELEASE file	**(0)=no**, 1=yes
20	IFLUX	Output of mass fluxes through output grid box boundaries (northward, southward, eastward, westward, upward and downward)	**(0)=off**, 1=on

Table 8. Continued.

21	MDOMAINFILL	Switch for domain-filling calculations: particles are initialized to reproduce air density or stratospheric ozone density; for limited-area simulations, particles are generated at the domain boundaries	**(0)=no**, 1 like air density, 2 stratospheric ozone tracer
22	IND_SOURCE	Unit to be used at the source; see Seibert and Frank (2004); Eckhardt et al. (2017)	**(1)=mass**, 2=mass mixing ratio
23	IND_RECEPTOR	Unit to be used at the receptor; see Seibert and Frank (2004); Eckhardt et al. (2017)	**(1)=mass**, 2=mass mixing ratio, 3=bwd. wet. dep., 4=bwd. dry. dep.
24	MQUASILAG	Quasi-Lagrangian mode to track individual numbered particles	**(0)=off**, 1=on
25	NESTED_OUTPUT	Switch to produce output also for a nested domain	**(0)=no**, 1=yes
26	LINIT_COND	Switch to produce output sensitivity to initial conditions given in concentration or mixing ratio units (in backwards mode only)	**(0)=no**, 1=mass concentration, 2=mass mixing ratio
27	SURF_ONLY	Output of SRR for fluxes only for the lowest model layer, most useful for backward runs when LINIT_COND set to 1 or 2	**(0)=no**, 1=yes
28	CBLFLAG	Skewed rather than Gaussian turbulence in the convective ABL; when turned on, very short time steps should be used (see CTL and IFINE)	**(0)=no**, 1=yes
29	OHFIELDS_PATH	Default path for OH file	
30	d_trop	Tropospheric horizontal turbulent diffusivity D_h	(**50 m^2 s^{-1}**)
31	d_strat	Stratospheric vertical turbulent diffusivity D_z	(**0.1 m^2 s^{-1}**)

Table 9. Contents of the user input file `RELEASES`.

Variable name	Description	Format, valid values, variable type
	Header (written only once and valid for all releases)	
NSPEC	Total number of species	Integer number
SPECNUM_REL	Species numbers in dir. `SPECIES`	Integer array of size NSPEC
	For each release	
IDATE1	Release start date	YYYYMMDD: YYYY=year, MM=month, DD=day
ITIME1	Release start time in UTC	HHMISS: HH hours, MI=minutes, SS=seconds; integer
IDATE2	Release end date	Same format as IDATE1
ITIME2	Release end time in UTC	Same format as ITIME1
LON1	Left longitude of release box	$-180 <$ LON1 < 180, or according to input winds; real
LON2	Right longitude of release box	Same format as LON1; real
LAT1	Lower latitude of release box	$-90 <$ LAT1 < 90, or according to input winds; real
LAT2	Upper latitude of release box	Same format as LAT1; real
ZKIND	Reference level	1: meters above ground, 2: meters above sea level, 3: pressure (hPa); integer
Z1	Lower height of release box	Meters above reference level; real
Z2	Upper height of release box	Meters above reference level; real
PARTS	Total number of particles to be released	Integer ≥ 1
	For each species (NSPEC times)	
MASS	Total mass emitted	in, e.g., kilograms or unitless for mixing ratio; 1 in backward mode; real
COMMENT	Comment	40-character string (e.g., name of release point)

Notice that the format of the SPECIES_nnn files has changed from previous FLEXPART versions and users need to update their files accordingly. The use of SPECIES_nnn files from older FLEXPART versions may lead to run time errors or erroneous results.

The following specifies the parameters associated with each physicochemical process simulated.

- Radioactive or chemical decay: set with pdecay; off if pdecay<0.
- Wet deposition for gases: set with pweta_gas, pwetb_gas (for below-cloud) and phenry (for in-cloud). Switch off for both in- and below-cloud if either pweta_gas or pwetb_gas is negative.
- Wet deposition for aerosols: set with pccn_aero, pin_aero for in-cloud scavenging and pcrain_aero, pcsnow_aero and pdquer for below-cloud scavenging.
- Dry deposition for aerosols: set with pdensity, pdquer and psigma; off if pdensity< 0.
- Dry deposition for gases: set with phenry, pf0 and preldiff; off if preldiff< 0. Alternatively, a constant dry deposition velocity pdryvel can be given.
- Settling of particles: set with pdensity and pdquer.
- OH reaction: chemical reaction with the OH radical can be turned on by giving parameter pohcconst (cm^3 $molecule^{-1}$ s^{-1}), pohdconst (K) and pohnconst (no unit) positive values; defined by Eq. (13).
- Emission variation: emission variation during the hours (local time) of the day and during the days of the week can be specified. Factors should be 1.0 on average to obtain unbiased emissions overall. The area source factors (useful, e.g., for traffic emissions) are applied to emissions with a lower release height below 0.5 m above ground level (a.g.l.) and the point source factors (useful, e.g., for power plant emissions) to emissions with a lower release height than 0.5 m a.g.l. Default values are 1.0.

5.1.4 File OUTGRID

The OUTGRID file specifies the domain and grid spacing of the three-dimensional output grid. Note that in a Lagrangian model, the domain and resolution of the gridded output are totally independent from those of the meteorological input (apart from the fact that the output domain must be contained within the computational domain). The OUTGRID file contains a namelist &OUTGRID specifying all parameters. The variables read in for this file and all the following input files have not changed in recent FLEXPART versions; thus, for further explanation, see Stohl et al. (1995). Example files can be found in the options directory in the FLEXPART distribution.

5.1.5 File OUTGRID_NEST

Output can also be produced on one nested output grid with higher horizontal resolution, defined in the file OUTGRID_NEST, but with the same vertical resolution as given in OUTGRID. The OUTGRID_NEST file contains a namelist &OUTGRIDN specifying all parameters.

5.1.6 File AGECLASSES

The option to produce age class output can be activated in the COMMAND file. The file AGECLASSES then allows for the definition of a list of times (in seconds, in increasing order) that define the age classes used for model output. With this option, the model output (e.g., concentrations) is split into contributions from particles of different age, defined as the time passed since the particle release. Particles are dropped from the simulation once they exceed the maximum age, allowing their storage locations to be reused for new particles. This is an important technique to limit the memory usage for long-term simulations. Thus, even if the user is not interested in age information per se, it may often be useful to set one age class to define a maximum particle age.

5.1.7 File RECEPTORS

In addition to gridded model output, it is also possible to define receptor points. With this option output can be specifically produced for certain points at the surface in addition to gridded output. The RECEPTORS file contains a list with the definitions of the receptor name, longitude and latitude. If no such file is present, no receptors are written to output.

5.1.8 Static data input files

Several files contain static input data that are not usually modified by the user. These are (by default) also located in the options directory. If modeling a species requires calculating OH reactions, an OH field stored in file OH_variables.bin needs to be present. The file IGBP_int1.dat is a land cover inventory; file surfdata.t gives the roughness length and leaf area index of the different land cover types, and file surfdepo.t contains surface resistances for dry deposition calculations.

5.2 Meteorological data and preprocessing routines

FLEXPART can be run with meteorological input data for global domains or for smaller, limited-area domains. The FLEXPART computational domain always corresponds to this mother domain set by the input data, while the output domain can be smaller. FLEXPART can also ingest

Table 10. FLEXPART variables set in the user input file `SPECIES_nnn` for species number *nnn*. Note that the variable names given in the input namelist are the same as used subsequently in FLEXPART but with a prepended letter p (for parameter). For instance, `pspecies` corresponds to `species`.

	Variable name	Description	Unit	Type
1	`pspecies`	Tracer name	no units	string
2	`pdecay`	Species half-life for radioactive or chemical decay; off if `pdecay<0`	seconds	real
3	`pweta_gas`	Gases wet deposition: below-cloud scavenging parameter A (for precip. of $1\,\mathrm{mm\,h^{-1}}$)	$1\,\mathrm{s^{-1}}$	real
4	`pwetb_gas`	Gases wet deposition: below-cloud scavenging parameter B (dependency on precip. rate)	1	real
5	`pcrain_aero`	Aerosols wet deposition: below-cloud scavenging rain collection efficiency moderator for rain C_{rain}	1	real
6	`pcsnow_aero`	Aerosols wet deposition: below-cloud scavenging rain collection efficiency moderator for snow C_{snow}	1	real
7	`pccn_aero`	Aerosols wet deposition: in-cloud scavenging, cloud condensation nuclei efficiency $\mathrm{CCN_{eff}}$	1	real
8	`pin_aero`	Aerosols wet deposition: in-cloud scavenging, ice nuclei efficiency $\mathrm{IN_{eff}}$	1	real
9	`preldiff`	Gases dry deposition: ratio $D = D_{\mathrm{H_2O}}/D_i$, $D_{\mathrm{H_2O}}$ of the diffusivity of H_2O to the diffusivity of the component D_i (the diffusivity of the species in the `SPECIES_nnn` file); dry deposition of gases is switched off by negative D	1	real
10	`phenry`	Gases dry deposition and in-cloud scavenging: Henry's constant H	$\mathrm{M\,atm^{-1}}$	real
11	`pf0`	Gases dry deposition: reactivity factor for oxidation of biological substances relative to that of ozone; $(0 \leq f_0 \leq 1)$ For nonreactive species f_0 is 0, for slightly reactive species it is 0.1 and for highly reactive species it is 1	1	real
12	`pdensity`	Aerosols dry deposition and settling: particle density ρ	$\mathrm{kg\,m^{-3}}$	real
13	`pdquer`	Aerosol dry deposition, aerosol wet deposition: below-cloud scavenging: particle mean diameter $\bar{d}$ Decides whether its gas(<=0) or aerosol (> 0)	m	real
14	`psigma`	Aerosol dry deposition: species diameter normalized deviation ($\sigma > 1$)	1	real
15	`pdryvel`	Gases dry deposition: dry deposition velocity (only used if `preldiff` and `pdensity` < 0)	$\mathrm{m\,s^{-1}}$	real
16	`pweightmolar`	Gases: species molar weight, used for volume mixing ratio (pptv) output	$\mathrm{g\,mol^{-1}}$	real
17	`pohcconst`	Gases OH reaction: C	$\mathrm{cm^3\ molecule\,s^{-1}}$	real
18	`pohdconst`	Gases OH reaction: D	K	real
19	`pohnconst`	Gases OH reaction: N	1	real
20	`parea_hour`	Emission variation factor (area source) for hour of the day, starting with 00:00–01:00 local time, 24 values Local time from longitude, no correction for summer time	1	real
21	`parea_dow`	Emission variation factor (area source) for day of the week, starting with Monday, 7 values	1	real
22	`ppoint_hour`	Emission variation factor (point source) for hour of the day, starting with 00:00–01:00 local time, 24 values Local time from longitude, no correction for summer time	1	real
23	`ppoint_dow`	Emission variation factor (point source) for day of the week, starting with Monday, 7 values	1	real

higher-resolution meteorological input data in subdomains of the mother domain. Such nested data must be available for the exact same times as those for the mother domain, checked by FLEXPART by comparing the time stamps in the two `AVAILABLE(_NESTED)` files. There is no nesting in the vertical direction and the poles must not be contained in any nest. To automatically produce the `AVAILABLE(_NESTED)` files, a Python script is available from the FLEXPART website (https://flexpart.eu/wiki/FpInputMetMkavail, last access: 25 June 2018) that checks which input files are present and then creates this file in the required format.

Compilation of FLEXPART v10.4 produces a single executable that automatically detects whether the meteorological input data come from the ECMWF IFS or NCEP GFS and whether they are in GRIB-1 or in GRIB-2 format. Nev-

ertheless, certain parameters may need to be adapted in `par_mod.f90` to the size of the meteorological input files (array dimensions), and the input grid may need to be shifted relative to the output grid (parameter `nxshift`). In the following, we describe how meteorological input data appropriate for FLEXPART can be retrieved from the ECMWF and NCEP.

5.2.1 ECMWF data retrieval

ECMWF data can be comprised of analysis and/or forecast data from the operational IFS data stream or specific reanalysis projects. For operational data, the meteorological fields can currently have a maximal temporal resolution of 1 h (more frequent data are not available), a vertical resolution of 137 model levels and $0.1^\circ \times 0.1^\circ$ horizontal resolution on a regular latitude–longitude grid. Other ECMWF datasets are not available at such high horizontal resolution. For example, ERA-Interim reanalysis data (Dee et al., 2011) with $1^\circ \times 1^\circ$ latitude–longitude resolution and 60 vertical levels can be retrieved 3-hourly by mixing 6-hourly analysis and 3 h forecast fields, but higher resolution is not available from the standard archive. The new Copernicus reanalysis ERA5 provides 1-hourly analysis fields with 137 model levels and a horizontal resolution of 31 km (0.28125°). Notice that access to some datasets, in particular the operational forecasts, is restricted and requires specific access (https://www.ecmwf.int/en/forecasts/accessing-forecasts, last access: 23 June 2018). However, reanalysis data (https://software.ecmwf.int/wiki/display/WEBAPI/Available+ECMWF+Public+Datasets, last access: 23 June 2018) are publicly available.

The IFS is a global model that uses spectral representation with spherical harmonics for the dynamical part and a grid-point representation on a reduced Gaussian grid for the physical part. However, FLEXPART needs the input data on a regular latitude–longitude grid, and thus IFS data have to be preprocessed. With respect to the vertical coordinate system, the data need to be on the native ECMWF model levels (η levels), which are subsequently transformed within FLEXPART to a terrain-following vertical coordinate system.

As explained above, each ECMWF dataset has its own specific temporal and spatial resolution, and the meteorological parameters provided can be different from dataset to dataset. To produce meteorological GRIB files suitable for FLEXPART input from these different datasets, a software called `flex_extract` (current version 7.0.4) has been developed specifically for this purpose. In order to prepare the GRIB files from the ECMWF Meteorological Archival and Retrieval System (MARS; https://software.ecmwf.int/wiki/download/attachments/45759146/mars.pdf, last access: 24 June 2018), several retrieval requests using the MARS command language and some further processing steps are needed. Since all ECMWF datasets need to be handled differently and some may not even contain all information needed for FLEXPART, `flex_extract` has a focus on some of the most important ones for driving FLEXPART. These are, in particular, the reanalysis datasets ERA-Interim (Dee et al., 2011), CERA-20C (the coupled climate reanalysis of the 20th century; Laloyaux et al., 2018) and the latest reanalysis ERA5, as well as data from the operational IFS stream. Each file (one for each time step) prepared by `flex_extract` for FLEXPART consists of a set of model-level and surface data as a combination of analysis and forecast fields depending on availability. For example, certain variables such as precipitation may only be available in forecast fields, whereas other data are also contained in analysis fields; `flex_extract` seeks an optimum combination of such data. Note that some parameters are stored as time-accumulated fields in the ECMWF archives and `flex_extract` calculates the instantaneous fluxes out of them (e.g., precipitation fluxes). For more details on this process of de-accumulation, see Hittmeir et al. (2018). Since FLEXPART needs the pressure hybrid coordinate vertical velocity as used in the ECMWF model, an important feature of `flex_extract` is the computation of this parameter from the horizontal wind field (see Stohl et al., 2001) for ERA-Interim and for the years when it was not operationally archived in MARS (before 2009).

The ECMWF is a European intergovernmental organization that grants full access to its multi-petabyte MARS archive for their member and cooperating states. Users with a full-access account can run `flex_extract` v7.0.4 directly on ECMWF servers or via a local gateway server. This mode is also required to retrieve the most recent operational data from the ECMWF. Users from member or cooperating states interested in this mode should contact the computing representative from their national meteorological service to obtain an account. Users from other countries worldwide can self-register at the ECMWF for a public account to be able to retrieve the public datasets (i.e., most reanalysis products); `flex_extract` v7.0.4 makes use of the WebAPI (https://software.ecmwf.int/wiki/display/WEBAPI/ECMWF+Web+API+Home, last access: 24 June 2018) tool provided by the ECMWF to access the data from outside their systems. This tool can distinguish between public and member state users. Therefore, it is also a convenient option for member and cooperating state users who only need data older than a few days from the operational stream or reanalysis data. A full-access account to ECMWF servers is no longer needed in this case.

The `flex_extract` software v7.0.4 is a set of Python routines combined with a Fortran program for faster computation of grid transformations and vertical velocity calculation. A Python 2.7 interpreter with several common modules, such as NumPy and date–time, are required and usually included in the Anaconda distribution (https://www.anaconda.com/download/, last access: 25 June 2018). Additionally, a Fortran compiler, the ECMWF WebAPI tool, the GRIB-API or ecCodes module, and the Emoslib interpolation library have to be available. Note that the GRIB-

API (or ecCodes) module has to be available for Python as well as for Fortran. Installation instructions can be found at ECMWF websites directly or in the Software Installation Plan for `flex_extract`. Knowledge of Python, although helpful, is not necessary for using the retrieval scripts. A certain knowledge of the ECMWF dataset to be retrieved is useful to understand the composition of retrievals, but many basic examples of `CONTROL` files are provided in the `flex_extract` distribution. These `CONTROL` files determine the key parameters for the `flex_extract` MARS retrievals and can be adapted to change domain as well as spatial and temporal resolution. Even for these few parameters the user should check for availability upfront. For example, ERA-Interim data have a maximum grid resolution of $0.75° \times 0.75°$ and 6-hourly temporal resolution for the public dataset.

The `flex_extract` v7.0.4 software is included in the FLEXPART v10.4 file tree under the directory `preprocess` (see Table 1). It can also be downloaded from https://flexpart.eu/ (last access: 30 October 2019) (https://flexpart.eu/downloads/62, last access: 30 October 2019) as a tarball. For more details the reader is referred to the `flex_extract` v7.0.4 user documentation (e.g., the Software Installation Plan – SIP.pdf – and the Software User Tutorial – SUT.pdf) in `preprocess/flex_extract`.

5.2.2 NCEP data retrieval

Meteorological data from the NCEP GFS are freely available, easily accessible and ingested by FLEXPART on pressure levels, unlike ECMWF data. These pressure-level data have lower resolution than model-level data but offer the advantage of great consistency between different datasets. Therefore, preprocessing NCEP data is much simpler than ECMWF data and limited to precipitation data, which are available only in forecast fields.

Both operational analysis data and several reanalysis datasets are available. Notice that NCEP also provides forecast data for free, which are not available from the ECMWF even for member state users except for national meteorological services or users with a special contract. The data retrieval from NCEP is described in a wiki page on the FLEXPART website (https://flexpart.eu/wiki/FpInputMetGfs, last access: 8 July 2018), where a script for downloading NCEP data can also be found. Operational GFS data can be downloaded by simple FTP or wget from a rolling archive of the meteorological forecast and analysis data (http://www.ftp.ncep.noaa.gov/data/nccf/com/gfs/prod/, last access: 8 July 2018) under the catalog `gfs.YYYYMMDDHH`, which contains fields in GRIB-2 format. Six-hourly NCEP FNL (Final) Operational Model Global Tropospheric Analyses (http://rda.ucar.edu/datasets/ds083.2/, last access: 30 October 2019) are available in near-real time since July 1999. These data are similar to the operational analyses, but NCEP also ingests late-incoming observation data for their production. Archived reanalysis datasets are also available from NCEP, e.g., the Climate Forecast System Reanalysis (CFSR) Selected Hourly Time-Series Products (http://rda.ucar.edu/datasets/ds093.1/, last access: 30 October 2019) for the period January 1979 to March 2011.

6 FLEXPART output

6.1 Output files overview

In the following we describe the FLEXPART output files together with changes made since the last documented FLEXPART version (Stohl et al., 2005). An overview of all possible output files is provided in Table 11. Notice that not all these files are written out in every model run; the user settings control which files are produced. At the beginning of a run, FLEXPART records descriptive metadata in the binary file `header`. This information is also written into the plain text files `header_txt` (with the exception of the orography data and release information). The release information is written in `header_txt_releases`. Corresponding files `header_nest` are produced if nested output is selected.

At each output time, FLEXPART produces files containing the gridded output. Separate files are created for every species and domain (mother and, if requested, nest). The naming convention for these files is `grid_type_date_nnn`. For forward runs, `type` can be `conc` or `pptv` for concentrations and mixing ratios or `flux` for 3-D mass fluxes across the grid cell faces (Stohl et al., 2005, Sect. 8.5). For backward runs, `type` can be `time` for the sensitivity of receptor concentrations to emission fluxes, `drydep` for the sensitivity of receptor dry deposition to emissions or `wetdep` for the sensitivity of receptor wet deposition to emissions. For backward runs, there can also be an output file `grid_initial_nnn`, which gives the receptor sensitivity to initial conditions; `date` denotes the date and time for which the output is valid, and `nnn` is the species number as specified in `RELEASES`. The list of the output times is progressively written to the text file `dates`. For the nested output, `grid` is replaced by `grid_nest`.

Wet and dry deposition fields in forward runs are calculated on the same horizontal output grid and are appended to `grid_conc_date_nnn` and `grid_pptv_date_nnn` files. The deposited matter is accumulated over the course of a model run. It generally increases with model time, but for species with radioactive decay, losses are possible. As for long simulations small deposition amounts may be added to already large deposited quantities, the default precision of the deposition fields was changed from single (in older FLEXPART versions) to double precision to avoid numerical inaccuracies when deriving instantaneous fluxes from accumulated quantities.

For a list of points at the surface, concentrations or mixing ratios in forward simulations can also be calculated indepen-

Table 11. List of FLEXPART output files and for which user settings ("switches") they are produced. See Table 12 for details on the units in the gridded output.

Name	Format	Switches	Description of contents
header	binary	default output	run metadata + ancillary data
header_txt	text	default output	human-readable run metadata (from COMMAND)
header_txt_releases	text	default output	human-readable run metadata (from RELEASES)
dates	text	default output	time series: dates of output files
grid_conc_date_nnn	binary (sparse array)	LDIRECT=1 IOUT=1,3,5	3-D tracer mass density + 2-D deposition
grid_pptv_date_nnn	binary (sparse array)	LDIRECT=1 IOUT=2,3	3-D tracer volume mixing ratio + 2-D deposition
grid_time_date_nnn	binary (sparse array)	LDIRECT=-1 IOUT=1	3-D sensitivity of atmospheric receptor to emissions
grid_drydep_date_nnn	binary (sparse array)	LDIRECT=-1 IOUT=1, IND_RECEPTOR=3	3-D sensitivity of dry deposition receptor to emissions
grid_wetdep_date_nnn	binary (sparse array)	LDIRECT=-1 IOUT=1, IND_RECEPTOR=4	3-D sensitivity of wet deposition receptor to emissions
grid_conc_date_nnn.nc	binary (NetCDF)	LDIRECT=1 IOUT=9,11,13	3-D tracer + 2-D wet and dry deposition
grid_time_date_nnn.nc	binary (NetCDF)	LDIRECT=-1 IOUT=9	3-D sensitivity of atmospheric receptor to emissions
grid_drydep_date_nnn.nc	binary (NetCDF)	LDIRECT=-1 IOUT=9, IND_RECEPTOR=3	3-D sensitivity of dry deposition receptor to emissions
grid_wetdep_date_nnn.nc	binary (NetCDF)	LDIRECT=-1 IOUT=9, IND_RECEPTOR=4	3-D sensitivity of wet deposition receptor to emissions
grid_initial_nnn	binary (sparse array)	LDIRECT=-1, LINIT_COND>0	3-D sensitivity of receptor concentrations and deposition to initial conditions
partposit_date	binary	IPOUT=1,2	particle positions and meteorological data
		IPOUT=1,2 MQUASILAG=1	particle positions and meteorological data numbered consecutively
partposit_average_date	binary	IPOUT=3	time-averaged particle positions and meteorological data
trajectories.txt	text	IOUT=4, 5	clustered trajectories
receptor_conc	binary	LDIRECT=1 IOUT=1,3,5,9,11,13	mass density at receptors
receptor_pptv	binary	LDIRECT=1 IOUT=2,3,10,11	volume mixing ratio at receptors
header_nest	binary	NESTED_OUTPUT=1	nest metadata + ancillary data
grid_conc_nest_date_nnn	binary (sparse array)		
grid_pptv_nest_date_nnn	binary (sparse array)	as for mother grid	as for mother grid in a
grid_time_nest_date_nnn	binary (sparse array)	+ NESTED_OUTPUT=1	higher-resolution latitude–longitude grid
grid_drydep_nest_date_nnn	binary (sparse array)		
grid_wetdep_nest_date_nnn	binary (sparse array)		
grid_conc_nest_date_nnn.nc	binary (NetCDF)		
grid_time_nest_date_nnn.nc	binary (NetCDF)		
grid_drydep_nest_date_nnn.nc	binary (NetCDF)		
grid_wetdep_nest_date_nnn.nc	binary (NetCDF)		

dently from the grid using a kernel method and recorded in the files `receptor_conc` and/or `receptor_pptv`.

If the particle dump option is activated, in addition to the gridded output, the particle coordinates together with additional variables such as pressure, humidity, density, tropopause height, ABL height and orography height are recorded in the binary files `partposit_date`. These data can be useful for a variety of different purposes, for instance diagnostics of the water cycle (Stohl and James, 2004). FLEXPART version 10.4 also has the new option to write out time-averaged particle positions and meteorological data. These are recorded in the files `partposit_average_date`. Such output may be useful to obtain, for instance, more representative heights for particles in the boundary layer, where particle positions change rapidly and this is not sampled sufficiently with instantaneous output. If plume trajectory mode is activated, for every release the positions of trajectory clusters representing the centers of mass of all released particles are recorded in the file `trajectories.txt` (Stohl et al., 2002, 2005, Sect. 10).

The physical unit used for the output data in the files `grid_conc_date_nnn` and `grid_time_date_nnn` depends on the settings of the switches `ind_source` and `ind_receptor`, following Table 12. It is noteworthy that the unit of mass mixing ratio can also be used in `grid_conc_date_nnn`. For forward runs, additional files `grid_pptv_date_nnn` can be created (setting `IOUT` to values of 2 or 3), which contain data such as volume mixing ratios (requires molar weight in `SPECIES_nnn` file). Source–receptor relationships (i.e., emission sensitivities) in backward mode for atmospheric receptors are written out in `grid_time_date_nnn` files; those for deposited mass are recorded in files `grid_wetdep_date_nnn` and `grid_drydep_date_nnn` (see Seibert and Frank (2004), Eckhardt et al. (2017), Sect. 2.5, and Table 12 for output units). Notice that the user can also provide different input units. For instance, if emissions in a forward run are specified in Becquerels (Bq), the output would be in nanobecquerels per cubic meter ($\mathrm{nBq\,m^{-3}}$) with `ind_source=1` and `ind_receptor=1`. Notice further that all gridded output quantities in FLEXPART are grid cell averages, not point values.

6.2 Sparse matrix output

Depending on the type of model run, the gridded output can contain many grid cells with zero values (e.g., dispersion from a point source, backward run from a single receptor). The output is therefore written in a sparse matrix format, which is specific to FLEXPART. The array containing the data to be written out is scanned for sequences of nonzero values. The number of sequences found is stored in an integer variable `sp_count_i`, and the field positions at which each sequence begins are stored in a 1-D integer array, `sparse_dump_i`, using a one-dimensional representation of the output field. The total number of nonzero values is stored in `sp_count_r` and the nonzero values themselves in the real vector `sparse_dump_r`. Since all physical output quantities of FLEXPART are greater than or equal to zero, nonzero sequences are stored in `sparse_dump_r` with alternating signs, which allows for the separation of different sequences upon reading. Finally, all four variables are written out to the unformatted output file. This format replaces the compression used up to version 7 (the smallest of a full dump and a simple sparse matrix format), saving up to 60 % of disk space. The sparse matrix data can be read, for example, with the functions `readgrid.f` (Fortran) and `flex_read.m` (MATLAB) described in Sect. 6.4

6.3 NetCDF output

FLEXPART v10.4 can also support output in NetCDF format if the NetCDF libraries are available. To activate NetCDF support, append `ncf=yes` to the `make` command. If FLEXPART is compiled and linked to the NetCDF libraries, output files in NetCDF format can be produced by adding 8 to the `IOUT` parameter in the input file `COMMAND`, e.g., `IOUT=9` corresponds to `IOUT=1` with the standard binary output; see Table 11 and Sect. 5.1.1. In the NetCDF module `netcdf_output_mod.f90` a parameter `write_releases` determines at compile time if the information on the releases should also be written to the NetCDF file. Only one NetCDF file is written that contains all species and all time steps. Both mother and nested output (if present) are contained in that file. Since the NetCDF output is specified in the climate and forecast (CF) format, any standard software can be used for displaying and processing the output (e.g., panoply, ncview). NetCDF output data files are compressed.

The NetCDF output file contains information on the run settings and the simulation grid from the `COMMAND` and `OUTGRID*` files. It also contains additional information in the header on the producing center, as listed in Table 13. The content of these attributes can be adapted in the file `netcdf_output_mod.f90` before compilation.

6.4 Post-processing routines

For the NetCDF output of FLEXPART, standard visualization tools, for example Panoply, can be used. For the sparse matrix binary output, several post-processing routines (MATLAB, Fortran, R, Python and IDL) have been developed in order to assist in the usage and analysis of these data. A number of post-processing tools are available online (https://flexpart.eu/wiki/FpOutput, last access: 16 August 2018). Note that some of these tools require reading a text string containing the model version. Since the length of this string changed in FLEXPART v9.2, the post-processing routines now require the allocation of a longer string.

Table 12. Physical units of the input (in file RELEASES) and output data for forward (files grid_conc_date_nnn) and backward (files grid_time_date_nnn) runs for the various settings of the unit switches ind_source and ind_receptor (for both switches, 1 refers to mass units, 2 to mass mixing ratio units). IOUT is 1 (or 9 for NetCDF output) except where indicated; "(dep.)" in lines 5 and 6 of the table refer to the deposition output provided in addition to the atmospheric output in files grid_conc_date_nnn.

Direction	File name	ind_source	ind_receptor	Input unit	Output unit
Forward	grid_conc*	1	1	kg	ng m^{-3}
	grid_conc*	1	2	kg	ppt by mass
	grid_conc*	2	1	1	ng m^{-3}
	grid_conc*	2	2	1	ppt by mass
	grid_conc*	1	1 or 2 (dep.)	kg	ng m^{-2}
	grid_conc*	2	1 or 2 (dep.)	1	ng m^{-2}
	grid_pptv* (IOUT=2, 3)	1	1	1	ppt by volume
Backward	grid_time*	1	1	1	s
	grid_time*	1	2	1	s m^3 kg^{-1}
	grid_time*	2	1	1	s kg m^{-3}
	grid_time*	2	2	1	s
	grid_wetdep*	1	3 (wet dep.)	1	m
	grid_drydep*	1	4 (dry dep.)	1	m
	grid_wetdep*	2	3 (wet dep.)	1	kg m^{-2}
	grid_drydep*	2	4 (dry dep.)	1	kg m^{-2}
	grid_initial*	1	1	1	1
	grid_initial*	1	2	1	m^3 kg^{-1}
	grid_initial*	2	1	1	kg m^{-3}
	grid_initial*	2	2	1	1

Table 13. Additional information in the NetCDF output file as attributes.

Conventions	CF-1.6 (NetCDF CF convention identifier)
Title	FLEXPART model output (content title)
Institution	producer string "institution" set in netcdf_output_mod.f90
Source	creation string "flexversion" model output set in FLEXPART.f90
History	date string with login and host name
References	Stohl et al. (2005)

Fortran routines are available for download on the FLEXPART website with the subroutines readheader.f for reading the header and readgrid.f for reading the gridded binary fields. Analysis or plotting programs written in Fortran can call these subroutines.

There are also MATLAB tools working in a similar way as the Fortran routines, with flex_header.m for reading the header and flex_read.m for reading the data fields. If particle dumps were made, the MATLAB function readpart.m reads the corresponding data files (a similar Fortran code is also available).

The R programs available for post-processing FLEXPART output include routines to read the binary output in the grid_conc (or grid_pptv) and grid_time files and to plot maps. Routines are also available to plot trajectories on a map from the file trajectories.txt and to plot time series of concentrations (or mixing ratios) from the file receptor_conc (or receptor_pptv).

Several Python tools are available for reading FLEXPART data from release 8.0 and above. The module reflexible, available from the FLEXPART website and also at https://github.com/spectraphilic/reflexible (last access: 6 August 2018), enables the user to easily read and access the header and grid output data of the FLEXPART model runs. It provides a simple tool that facilitates consistent reading of both the original sparse matrix output files and the NetCDF output. Some basic plotting functionality is provided to quickly assess and validate runs or to look at the input parameters. An alternative Python tool is Quicklook that can be also downloaded from the https://flexpart.eu website.

7 Application examples

In this section we provide 38 examples of the FLEXPART model that serve three purposes: (1) verification of a new FLEXPART installation; (2) demonstration of the model ca-

pabilities for new users; and (3) confirmation of consistency in the model output when code changes are made that should not change the results. These examples do not represent an exhaustive set of all possible model uses, but they are designed to demonstrate and test different widely used functionalities of the model.

All examples are variations of a default example case, which uses the settings in the user input files as distributed with the FLEXPART v10.4 code package. These default input files are located in the directory `options` (Sect. 5) and are consistent with the default meteorological data retrieved from the ECMWF by the `flex_extract` package (Appendices A5 and B1). An `AVAILABLE` file fitting with these input data is also distributed with FLEXPART. These default settings are described in detail in Appendix B2.

Using the default example as a basis, the different functionalities of the model can be activated by adequately changing certain parameters in the user input files, thereby generating 36 other example runs. We have categorized these examples into 10 different groups; each group explores different capabilities of the model. Table 14 lists all examples and the parameter changes needed to produce them. The first group includes the default example and explores the different options for producing gridded model output (e.g., output units, output formats) for a simple forward model run with a single starting point over the North Atlantic. The second group of examples introduces FLEXPART's backward simulation capability. The third group demonstrates different usages of the particle dump output. The fourth group gives examples for the use of mass vs. mass mixing ratio units at both the source and the receptor and for both forward and backward simulations to establish source–receptor relationships as in Seibert and Frank (2004). The fifth group produces output for different chemical species and aerosols. The sixth group illustrates the use of nested output fields. Group seven is constituted by a single domain-filling run, as used, for instance, in Stohl and James (2004). Group eight contains settings for a backward run providing 2-D sensitivities to gridded surface fluxes and 3-D sensitivities to initial conditions, as they are typically required for the inverse modeling of greenhouse gases (e.g., Thompson et al., 2017). Group nine shows the use of the new skewed turbulence parameterization (Cassiani et al., 2015). Group 10 shows the use of the new backward deposition (Eckhardt et al., 2017). Group 11 contains a forward 2 d run simulating instantaneous emissions from a hypothetical Grímsvötn eruption (Fig. 9).

The list of examples may be extended in the future to allow for the testing of even more model features and to provide a reference archive to see how FLEXPART results may change as the code is being developed further. The user can get these reference results from https://flexpart.eu. A quick reference containing mean and maximum grid values for every example is also referenced in Appendix B5.

The directory `tests/examples/` contains scripts that generate all the files necessary to run the examples. These

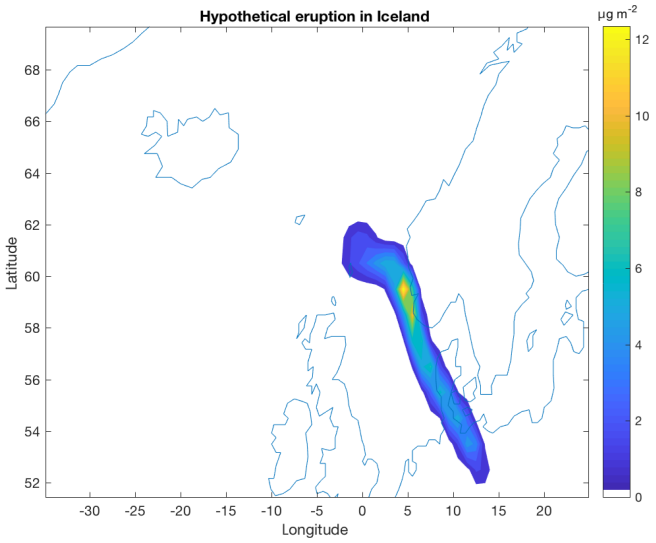

Figure 9. Hypothetical Grímsvötn eruption on 1 April 2015 at 00:00 UTC (instantaneous release). Total column concentrations are shown ($\mu g\,m^{-2}$) 18 h after the eruption.

scripts, described in Appendix B3, generate the input files by modifying the namelists in the default `options` directory provided with the distribution. This is done by the bash script `gen_options_all.sh`. For instance, the example "bwd" is generated by changing the line containing the parameter `LDIRECT` to -1 in the file `COMMAND`.

After the input data files are generated, all examples can be executed interactively from the command line. Alternatively, the script `gen_batch_jobs_cl.sh` generates a batch script for each case (to be run from the command line or using a workload manager such as SLURM). This procedure automates the sample output generation. Once the output files are created, they can be read using the tools in the directory `postprocess`. They can be plotted and analyzed with, e.g., the reading routines described in Sect. 6.4. In addition, some testing capabilities have been added. These are presented in Appendix B.

8 Final remarks, outlook and future code development

In this paper, we have described the Lagrangian particle dispersion model FLEXPART v10.4; 2 decades ago, the model code was developed mainly by one person, with specific code input from a few other researchers. At that time, no specific measures were needed to ensure code consistency, track code changes or identify coding bugs. However, as the number of FLEXPART users has grown substantially in recent years, more and more people have started to develop the code, contributed code snippets, and reported or identified bugs. The resulting code changes range from the adaptation of more modern coding standards, parallelization and efficiency enhancements, the improvement of the model functionality, and

Table 14. List of the test cases for FLEXPART 10.4.

Group	Test name	Change to default options	Description
(1) Gridded output	1	default (IOUT=1)	default option: forward run with mass concentration output; see Table 12
	2	IOUT=2	mixing ratio
	3	IOUT=3	concentration and mixing ratio
	5	IOUT=5	concentration and trajectory cluster
	9	IOUT=9	mass concentration in NetCDF output format
	10	IOUT= 10	mixing ratio in NetCDF output format
	11	IOUT= 11	concentration and mixing ratio in NetCDF output format
(2) Backward	bwd	LDIRECT= −1	SRR
	bwd5	LDIRECT=−1, IOUT=5	backward trajectory cluster
	bwd_nc	LDIRECT=−1, IOUT=9	SRR in NetCDF output format
(3) Particles	part1	IPOUT= 1	particle dump
	part2	IPOUT=2	particle dump at end of simulation
	part_bwd1	IPOUT=1, LDIRECT=−1	backward trajectories
(4) Units	ind_1_2	IND_RECEPTOR=2	receptor (gridded) in mass mixing ratio units
	ind_2_1	IND_SOURCE=2	source in mass mixing ratio units
	ind_2_2	IND_SOURCE=2, IND_RECEPTOR= 2	source and receptor in mass mixing ratio units
	bwd_ind_1_2	IND_RECEPTOR=2, LDIRECT=−1	receptor in mass mixing ratio units, backward
	bwd_ind_2_1	IND_SOURCE=2, LDIRECT=−1	source in mass mixing ratio units, backward
	bwd_ind_2_2	IND_SOURCE=2, IND_RECEPTOR=2, LDIRECT=−1	source and receptor in mixing ratio units, backward
(5) Species	specNO	SPECNUM_REL=3	nitric oxide species
	specCO	SPECNUM_REL=22	carbon monoxide species
	specAERO-TRACE	SPECNUM_REL=23	idealized aerosol simulation
	specBC	SPECNUM_REL=40	black carbon simulation
(6) Nests	nested	NESTED_OUTPUT= 1	nested output
	nested_mr	NESTED_OUTPUT=1, IOUT=2	volume mixing ratio nested output
	nested_bwd	NESTED_OUTPUT=1, LDIRECT=−1	nested output backwards
	nested_nc	NESTED_OUTPUT=1, IOUT=9	nested output in NetCDF format
	nested_mr_nc	NESTED_OUTPUT=1, IOUT=10	volume mixing ratio nested output NetCDF
	nested_bwd_nc	NESTED_OUTPUT=1, LDIRECT=−1, IOUT=9	nested output backwards in NetCDF output format
(7)	DOMAINFILL	MDOMAINFILL= 1	domain-filling run
(8) Init. cond.	init_cond	LINIT_COND=1, LDIRECT=−1	sensitivity to initial conditions
	init_cond_ind_1_2	LINIT_COND=1, LDIRECT=−1, IND_RECEPTOR=2	sensitivity to initial conditions in mixing ratio
	init_cond_surf	LINIT_COND=1, SURF_ONLY=1, LDIRECT=−1	sensitivity to initial conditions adapted to surface fluxes
(9) CBL	CBLFLAG	CBLFLAG=1, CTL=40, IFINE=5	skewed turbulence
	CBLFLAG_bwd	CBLFLAG= 1, LDIRECT=−1, CTL=40, IFINE=5	skewed turbulence backwards
(10) B.d.	bwd_ind_1_3	IND_RECEPTOR=3, LDIRECT=−1	backward wet deposition
	bwd_ind_1_4	IND_RECEPTOR=4, LDIRECT=−1	backward dry deposition
(11)	Volc	Modified `RELEASES` file	hypothetical volcanic eruption

the addition of output options, to revisions and extensions of the model physics. All this has been documented in this paper. Integration of all these changes into a single stable model version represents a growing challenge in itself, and efforts to address this challenge (e.g., model website and repository, version control, testing environment) have also been documented here.

As FLEXPART is developed further, updates will continue to be made available on the FLEXPART website at https://flexpart.eu. We encourage established and new users to contribute to FLEXPART development by providing their code changes, as well as a description of these changes, as new feature branches of the latest commits in the FLEXPART git repository. New code should pass all test cases provided in the FLEXPART distribution and provide consistent output, unless there are specific reasons why output should be different, such as improvements in the model physics. This will expedite the integration of important new model features in the main development branch of the model.

Code and data availability. The code described in this work is archived as flexpart10.4.tar at https://doi.org/10.5281/zenodo.3542278 (Pisso et al., 2019). FLEXPART downloads are available at https://www.flexpart.eu/downloads (last access: 22 November 2019) and FLEXPART releases are available at https://git.nilu.no/flexpart/flexpart/-/releases (last access: 22 November 2019). The working git repository for this version 10.4 (branches master, dev and release) can be accessed at https://git.nilu.no/flexpart/flexpart (last access: 22 November 2019) as well as the mirrors https://www.flexpart.eu/gitmob/flexpart (last access: 22 November 2019), https://github.com/flexpart/flexpart (last access: 22 November 2019) and https://bitbucket.org/flexpart/flexpart (last access: 22 November 2019).

Appendix A: Installing FLEXPART and `flex_extract`

Here, we provide step-by-step instructions on how to install FLEXPART on Linux from scratch. This has been tested on an Ubuntu 16.4 distribution running on a dedicated instance in the Amazon cloud. Notice that in most environments, some or all of the required libraries (e.g., a Fortran compiler) are already installed and an installation totally from scratch would thus not be needed. In such cases, we strongly recommend that these libraries are used instead of installing everything from scratch. However, sometimes it may be necessary to install them from source (e.g., to avoid incompatibilities between different compilers or different versions of the same compiler). In the following, we assume that the user has root privileges in the system, but it is also possible for normal users to install the libraries in nonstandard locations. It is possible to ask for help by writing to the FLEXPART user email list (registration needed) or by creating a ticket on the community website at https://flexpart.eu.

A1 System requirements

To begin, ensure that the latest packages are being used. This section is for completeness only, and most users (if not starting from a new system installation) can skip it and jump to Sect. A2.

```
sudo apt-get update
```

FLEXPART is developed using gfortran.

```
sudo apt-get install g++ gfortran
```

Some libraries (e.g., `grib_api`, `jasper-1.900.1`) require the GNU autotools suite in order to configure, build and install.

```
sudo apt-get install autoconf libtool
 automake flex bison
```

Newer packages (e.g., ecCodes) use CMake instead.

```
sudo apt-get install cmake
```

Python is not required for FLEXPART itself but is necessary for some preprocessing and post-processing tools, in particular `flex_extract` for retrieving ECMWF wind fields. Git is recommended to access the code repositories. An editor (e.g., vim) is usually also necessary.

```
sudo apt-get install python-dev
 python-pip git-core vim
```

A2 Installing GRIB libraries

If JPG compression is needed to decode the input meteorological winds, download the jasper library from the jasper project page (http://www.ece.uvic.ca/~mdadams/jasper/, last access: 30 October 2019) and install it.

```
curl https://www.ece.uvic.ca/~frodo/jasper/

software/jasper-1.900.1.zip
 --output jasper-1.900.1.zip
sudo apt install unzip
unzip jasper-1.900.1.zip
cd jasper-1.900.1
./configure
make
make check
sudo make install
```

Download the grib_api library from the ECMWF website (https://software.ecmwf.int/wiki/display/GRIB/Home, last access: 30 October 2019) and install it.

```
gunzip grib_api-X.X.X.tar.gz
tar -xf grib_api-X.X.X.tar
./configure [--prefix=grib_api_dir]
 [--with_jasper=<jasper installation
  path>]
make
make check
make install
```

If you have no root privileges in your system, give the full path of `grib_api_dir` to the prefix option. If jasper is in a nonstandard location, is has to be passed to the `grib_api` configuration script. Please note that GRIB-API is no longer maintained. The primary GRIB encoding–decoding package used at the ECMWF is currently ecCodes. Any new features for the handling of GRIB files will only be developed in ecCodes. However, for FLEXPART v10.4 `grib_api` is sufficient. We keep the `grib_api` instructions for backward consistency.

For future versions, ensure that the path `/usr/local/lib/` is in the environment variable `$PATH`; otherwise, ecCodes may not find it. Obtain and unpack ecCodes.

```
curl https://software.ecmwf.int/wiki/
download/attachments/45757960/
eccodes-2.7.3-Source.tar.gz \
--output eccodes-2.7.3-Source.tar.gz
tar -xvf eccodes-2.7.3-Source.tar.gz
```

The ecCodes requires CMake. The installation procedure is described on the ECMWF ecCodes web page.

A3 Installing NetCDF libraries

NetCDF output is optional. In order to enable NetCDF output, the NetCDF library has to be available in the system. For building the NetCDF library it is recommended to first build HDF5 with support for compression (zlib). For this, download zlib (version 1.2.8) from the zlib website (https://www.zlib.net/, last access: 30 October 2019) and install it.

```
tar -xzvf zlib-1.2.8.tar.gz
cd zlib-1.2.8/
./configure [--prefix=<installation path>]

make
make install
```

Download HDF5 from the HDF Group website (https://www.hdfgroup.org/HDF5/release/obtainsrc.html, last access: 30 October 2019) and install it.

```
tar -xzvf hdf5-1.8.17.tar.gz
cd hdf5-1.8.17/
./configure --with-zlib=<path to zlib>
 [--prefix=<installation path>]
make
make check
make install
```

Download the latest stable version of NetCDF–C from the Unidata website (https://www.unidata.ucar.edu/downloads/netcdf/, last access: 30 October 2019) and install it.

```
tar -xzvf netcdf-4.4.1.tar.gz
cd netcdf-4.4.1/
./configure --enable-netcdf-4
 [--prefix=<installation path>]
make
make check
make install
```

Download the latest stable version of NetCDF–Fortran from the Unidata website (https://www.unidata.ucar.edu/downloads/netcdf/, last access: 30 October 2019) and install it.

```
tar -xzvf netcdf-fortran-4.4.4.tar.gz
cd netcdf-fortran-4.4.4/
./configure [--prefix=<installation path>]

make
make check
make install
```

A4 Installing FLEXPART

Download the latest release of the FLEXPART source from the FLEXPART community website (https://flexpart.eu/wiki/FpDownloads, last access: 14 November 2019) or from https://transport.nilu.no (last access: 14 November 2019) and untar it.

```
tar -xvf flexpart10.4.tar.gz
```

Alternatively, clone the FLEXPART repository directly from the FLEXPART community site git.

```
git clone https://www.flexpart.eu/gitmob/

flexpart
```

This mirrors https://git.nilu.no/flexpart/flexpart. Additional mirrors exist, e.g., at Bitbucket (https://bitbucket.org/flexpart/flexpart, last access 30 October 2019) and GitHub (https://github.com/flexpart/flexpart, last access 30 October 2019). Edit the library path variable in the `makefile` according to the position of `libeccodes` (or `libgrib_api`) and `libjasper`. Optionally, edit the file `par_mod.f90` to set parameters for the meteorological data, grid dimension and maximum particle number (`maxpart`, `maxspec`, `nxmax`, `nymax`, `nuvzmax`, `nwzmax`, `nzmax`, `nxshift`). The default values are set to work with the test cases in Sect. 7 but may be too small for large simulations or too large for the available system resources. Then type

```
make
```

in order to create the executable. Invoking the executable `FLEXPART` should now print in the standard output.

```
Welcome to FLEXPART Version 10.4
FLEXPART is free software released under
 the GNU General Public License.
```

However, without access to valid input data, the program will issue an error. Appendix C explains how to generate valid output with the standard meteorological fields from the ECMWF that can be obtained following the procedure described in Sect. A5. The makefile also allows the following command.

```
make clean
```

This can be used to safely remove all object and module files, e.g., if one wants to recompile after compiler option changes.

A5 Installing `flex_extract`

A short description of the installation steps for this software is given for the public user mode (other modes are described in the `flex_extract` documentation). For this mode, the user does not need to be a member state user (https://www.ecmwf.int/en/about/who-we-are/member-states, last access: 13 October 2018) but can simply register at the ECMWF website. For the other operating modes and a more detailed explanation, see the `README.md` file of the `python` directory in the `flex_extract` distribution or the documentation files `SIP.pdf` and `SUT_ondemand.pdf`.

First of all, the user should register at the ECMWF website (https://www.ecmwf.int/en/forecasts/accessing-forecasts/order-historical-datasets, last access: 13 October 2018). To access public datasets each dataset license has to be

accepted separately before the account can be used for retrieval of these data. This can be done at the following website: https://software.ecmwf.int/wiki/display/WEBAPI/Available+ECMWF+Public+Datasets (last access: 13 October 2018).

A5.1 System preparation for `flex_extract`

`flex_extract` requires a Python environment and a Fortran compiler. See Sect. A1 for installation instructions. To prepare the environment for the `flex_extract` installation, it is advisable to consider the official documentation and information from the ECMWF websites. We recommend the following steps.

1. For important information read the `Emoslib` (https://software.ecmwf.int/wiki/display/EMOS/Emoslib, last access: 13 October 2018) installation instructions first.

2. Read the ECMWF blog about gfortran (https://software.ecmwf.int/wiki/display/SUP/2015/05/11/Building+ECMWF+software+with+gfortran, last access: 13 October 2018) for details on the installation process of the libraries.

3. Install `FFTW` (http://www.fftw.org, last access: 13 October 2018) for Fortran, which is a library for computing the discrete Fourier transformation. This library is necessary for `Emoslib`. (Note: apply `make` twice! Once without any options and once with single precision option; see the information on the `Emoslib` website).

4. Install the interpolation library `Emoslib` for Fortran.

5. Install `ecCodes` (https://software.ecmwf.int/wiki/display/ECC, last access: 13 October 2018) or `grib_api` (https://software.ecmwf.int/wiki/display/GRIB/Home, last access: 13 October 2018) (for Python and Fortran). The `grib_api` support will be discontinued at the end of 2018 but `ecCodes` is downward compatible with `grib_api`.

6. Install the ECMWF `WebAPI` (https://confluence.ecmwf.int//display/WEBAPI/Access+MARS, last access: 13 October 2018) client by following the instructions on the website. It is a Python library to provide external access to the ECMWF servers.

7. Check whether `LD_LIBRARY_PATH` and `PATH` environment variables contain all paths to the previously installed libraries. The user should modify the `.bashrc` or `.tcshrc` file to guarantee that the variables contain the paths every time a new console is used.

8. Install the python package `numpy` via pip (https://scipy.org/install.html, last access: 13 October 2018).

9. Check the availability of Python packages (e.g., check the Python console for the following commands: `import eccodes`, `import grib_api`, `import ecmwfapi`)

10. Start a simple test retrieval (following the instructions on the ECMWF `WebAPI` website).

11. Install `flex_extract` (see the next section).

It is important to use the same compiler and compiler version for all libraries and the Fortran program `CONVERT2`.

A5.2 Building `flex_extract`

To install `flex_extract` a script `install.py` was prepared. The user can find it in the `python` directory of the `flex_extract` distribution.

The public user mode requires a local installation of `flex_extract`. Hence, we recommend adapting the paths to `ecCodes`, `Emoslib` or `grib_api` in one of the prepared makefiles, such as `Makefile.local.gfortran`, which can be found in the `src` directory. If a different compiler is used, this must also be adapted in the makefile. Then the installation script can be called as follows.

```
./install.py --target=local
 --makefile=Makefile.local.gfortran
```

With this setting `flex_extract` is installed within the current `flex_extract` directory. To install it in a different place, e. g. within a `FLEXPART` distribution, the user can set the path with the parameter `flexpart_root_scripts`. The installation was successful if the compilation of the Fortran program (`CONVERT2`) did not fail and is displayed at the end in the terminal.

A5.3 Running `flex_extract`

`flex_extract` is controlled by providing `CONTROL` files that contain a list of parameter settings. These parameters are described in detail in the Software User Tutorial (`SUT.pdf`) in the `docs` directory. The `CONTROL` files specify which ECMWF dataset is to be retrieved, the time and spatial resolution, the format of the GRIB file, and other options. In the Python directory are some example `CONTROL` files for the different datasets and access modes. They can be used as templates. `CONTROL` files with a `.public` ending are usable for the public access mode. The main difference is the parameter `dataset`, which explicitly specifies the public datasets. Note that not all meteorological fields, times and parameters were archived in the public datasets. This is already considered in the public `CONTROL` files.

To run `flex_extract`, the main program `submit.py` must be called. It retrieves the ECMWF data and generates the FLEXPART input files. To show all possible parameter options one can use the `-h` option. The script must be called

from the python directory of the flex_extract distribution. From the -h output it is clear that most parameters have default values or were already set via a CONTROL file parameter, except for the date. To retrieve just one day, one only needs to provide the start date. The rest will be done by flex_extract. This leads to the following script call for an arbitrary date.

```
./submit.py --controlfile=CONTROL_EI.public \

            --start_date=20120101 \
            --public=1
```

The program now displays each MARS request and some messages for the preparation of the FLEXPART input files. Eventually, the program will finish with a Done! message if there was no error. Output will be stored in the default directory work, which is a subdirectory of the distribution directory (flex_extract_v7.0.4). The produced files can serve as input to FLEXPART.

Appendix B: Running and testing FLEXPART

After a working FLEXPART executable is built (Appendix A), the next step is running the model and generating valid output. This requires consistent meteorological input data and user input files. In this section we describe the following: how to obtain the necessary wind fields (1), how to test run the executable with a default example (2), how to generate other examples (3), and how to run these examples and compare them with a reference output (4). In the following, $flexhome indicates the path to the root FLEXPART directory (e.g., $HOME/flexpart/) and $flex_extracthome indicates the path to the flex_extract root directory (e.g., $flexhome/preprocess/flex_extract/).

B1 Meteorological input for the examples

Appendix A describes how to build the flex_extract version included in the source code. Here, we describe the settings to produce the meteorological input data required for running the default (Sect. B2) and derived (Sect. B3) cases. The instructions are for ECMWF ERA5 reanalysis, which is a publicly available dataset (https://confluence.ecmwf.int/display/WEBAPI/Access+ECMWF+Public+Datasets, last access: 30 October 2019). Therefore, the data can be obtained via ecmwfapi and no special access rights to the ECMWF are needed. However, in order to retrieve the data the user needs to register, obtain a personal Secure Shell (SSH) key and properly configure the file .ecmwfapirc. The execution of the retrieval requires the Python packages ecmwfapi (for access) and grib_api or eccodes (for processing). To retrieve the data, execute the following commands.

```
export PYTHONPATH=path/to/ecmwfapi:path/to/
 grib_api
$flex_extracthome/Python>./submit.py
 --start_date 20170102  --controlfile
   CONTROL_EA5
```

This should generate the files EA170102?? in the following directory.

```
$flex_extracthome/work/
```

An AVAILABLE consistent with these wind fields is shipped together with the FLEXPART distribution.

```
$flexhome/AVAILABLE
```

B2 Running the default example: installation verification

With the input files, which are included in the FLEXPART distribution and described in Sect. 5, a first test case to verify that FLEXPART was installed correctly can be run. To start the model run, the meteorological data have to be in $flex_extracthome/work/ (see Sect. B1), the file pathnames in $flexhome, and the executable in $flexhome/src/ in the $flexhome directory type.

```
$flexhome>./src/FLEXPART
```

The results created by this run are stored, e.g., in a directory $flexhome/output (as defined in pathnames). The run should end with the following message.

```
CONGRATULATIONS: YOU HAVE SUCCESSFULLY
 COMPLETED A FLEXPART MODEL RUN!
```

If this message is received, the model has completed the simulation, which confirms that FLEXPART and all required libraries are installed correctly. However, it does not guarantee valid output. To verify that the results obtained are valid, see Sect. B5.

B3 Generating variations of the default example

To demonstrate more functionalities, a set of shell scripts generating different FLEXPART setups are provided in $flexhome/tests/examples. The script set_default_example.sh takes the content of the options directory and pathnames file from Sect. B2 as a basis, and then gen_options_all.sh creates new options_suffix directories for all of the cases described in Table 14. Here, the suffix corresponds to the example name as given in column 2 in Table 14. Finally, the script gen_pathnames.sh generates corresponding pathnames_suffix files pointing to all the options_suffix directories. With this, all example cases in Table 14 are ready to run.

B4 Running the examples

The examples can be run interactively one by one by invoking FLEXPART with the corresponding `pathnames_suffix` file. Alternatively, the script `gen_batch_jobs_cl.sh` generates a one-line script for each example case containing a call of FLEXPART and the appropriate `pathnames_suffix` file as a command-line parameter. All example scripts can then be run sequentially with `run_batch_cl.sh`, which creates `output_suffix` directories with the results, as well as log files `batch_job_pathnames_suffix.stdout` for each run. The examples described above can now be read and plotted with the tools included in the distribution. These steps are also automated in a `makefile`. All of the files and directories created by executing the scripts from Sects. B2 to B4 can be removed again with the command `make clean`.

B5 Comparing the results

To verify that FLEXPART is producing valid output, it is useful to compare the output of a new installation with existing model output. It is also useful to repeat such a comparison after code changes to make sure the output is not affected, except for model simulations in which changes in the results are intended. While comprehensive comparisons of model results are possible, here we provide only a very simple way of checking the model results. The directory included in the FLEXPART distribution `$flexhome/tests/examples_reference/` contains the output of the examples described in Table 14. The file `read_examples_output.txt` contains, for the relevant examples that produce gridded output, the mean and the maximum value that occurs in the gridded output files. This shall serve as a reference to which users can compare their results and thus verify that the model produces output as expected. In addition, the directory `compare_examples` contains the script `compare_grids.sh` that allows for the partial automation of this task (output in `compare_examples_output.txt`).

Appendix C: FLEXPART model versions

In addition to the reference version of FLEXPART described in this paper, there are many different model branches that were developed either for special purposes or to ingest other meteorological input data. This Appendix provides an incomplete list and a short description of some of these other versions. Further contributions are welcome in order to keep this list up to date.

C1 FLEXPART–NorESM/CAM

Recently, the FLEXPART model version FLEXPART–NorESM/CAM was developed, which is tailored to run with the meteorological output data generated by the CMIP5 version of NorESM1-M (the Norwegian Earth System Model) with $1.89° \times 2.5°$ horizontal resolution and 26 vertical levels. The standard time resolution of the NorESM/CAM meteorological data is 3 h. FLEXPART–NorESM/CAM is based on FLEXPART v9, and the atmospheric component of NorESM1-M is based on CAM4 (the Community Atmosphere Model). The adaptation of FLEXPART to NorESM required new routines to read meteorological fields, new post-processing routines to obtain the vertical velocity in the FLEXPART coordinate system and other changes, as detailed by Cassiani et al. (2016). The code can be downloaded from https://www.flexpart.eu/wiki/FpClimateNorESM (last access: 30 October 2019).

C2 FLEXPART–WRF

This FLEXPART version uses output from the Weather Research and Forecasting (WRF) mesoscale meteorological model (Brioude et al., 2013). Originally it was developed at the PNNL (Pacific Northwest National Laboratory) and named PILT (PNNL Integrated Lagrangian Transport). Compared to PILT, the further developed FLEXPART–WRF can use both instantaneous and time-averaged meteorological output of the WRF model. The latest version also includes the skewed turbulence scheme that was subsequently ported to the standard FLEXPART version 10.4. FLEXPART–WRF output can either be in binary or Network Common Data Form (NetCDF) format, both of which have efficient data compression. FLEXPART–WRF also offers effective parallelization with OpenMP in shared memory and an MPI library in distributed memory. Released versions of the code can be downloaded from https://www.flexpart.eu/wiki/ (last access: 30 October 2019) or cloned from the open repository git@git.nilu.no:flexpart/flexpart-wrf.git.

C3 FLEXPART–COSMO

In Europe several national weather services and research groups cooperate to develop and operate the non-hydrostatic limited-area atmospheric model COSMO (Consortium for Small-scale Modeling). At MeteoSwiss COSMO is operationally run with data assimilation on two grids with approximately $7 \times 7\,\mathrm{km}^2$ and $2 \times 2\,\mathrm{km}^2$ horizontal resolution centered over Switzerland. This enables the study of atmospheric transport over complex terrain on a long-term basis. To this end, we have developed a new version of FLEXPART that is offline coupled to COSMO output (FLEXPART–COSMO hereafter) and supports output from multiple COSMO nests. Particles are internally referenced against the native vertical coordinate system used in COSMO and not, as in standard FLEXPART, in a terrain-following z system. This eliminates the need for an additional interpolation step. A new flux deaccumulation scheme was introduced that removes the need for additional preprocessing of the input files. In addition to

the existing Emanuel-based convection parameterization, a convection parameterization based on the Tiedtke scheme, which is identical to the one implemented in COSMO itself, was introduced. A possibility for offline nesting of a FLEXPART–COSMO run into a FLEXPART–ECMWF run for backward simulations was developed that only requires minor modifications of the FLEXPART–ECMWF version and allows particles to leave the limited COSMO domain. The OpenMP shared memory parallelization to the model allows for asynchronous reading of input data. The code is available on request from dominik.brunner@empa.ch and stephan.henne@empa.ch.

C4 FLEXPART–AROME

The Applications of Research to Operations at Mesoscale (AROME) numerical weather prediction model is run operationally by Météo-France at the mesoscale. AROME forecasts for Europe exist at a resolution ranging from 0.5 to 2.5 km. The standard time resolution of the AROME meteorological data is 1 h. Based on FLEXPART–WRF, a coupling between FLEXPART and AROME was developed at Laboratoire de l'Atmosphère et des Cyclones (LACy, a joint institute between CNRS, Météo-France and the University of Reunion Island) using AROME high-resolution ($2.5 \times 2.5\,km^2$) forecasts over the southwest Indian Ocean. The FLEXPART–AROME branch (Verreyken et al., 2019b) simulates turbulent transport using the Thomson turbulent scheme (Thomson, 1987) already implemented by Lin et al. (2003) in the Stochastic Time-Inverted Lagrangian Transport (STILT) model. This method constrains mass transport between different turbulent regions to conserve mass locally for a passive well-mixed tracer. Turbulent kinetic energy profiles are taken directly from AROME model outputs. Such treatment of turbulent motion ensures consistency between the turbulence in the meteorological fields calculated by the NWP model and turbulence computed in the offline Lagrangian transport model. It has been noticed that the use of a dedicated ABL scheme such as Hanna in the FLEXPART model may generate inconsistency between the ABL turbulent domain and the resolved wind fields used to drive FLEXPART. Simulations using the Thomson scheme show a better representation of the turbulent mixing between boundary layer air and free tropospheric air.

C5 TRACZILLA

This branch-off from FLEXPART version 5 was originally developed for studies of transport and mixing in the upper troposphere–lower stratosphere region (e.g., Legras et al., 2003; Pisso and Legras, 2008). The modifications from the FLEXPART advection scheme consist mainly of discarding the intermediate terrain-following coordinate system and performing a direct vertical interpolation of winds, linear in log pressure, from hybrid levels. The vertical velocities are computed by the FLEXPART preprocessor using a mass-conserving scheme in the hybrid ECMWF coordinates. Alternatively, the vertical velocities can be computed from the rates of diabatic heating from ECMWF winds. In addition to the reanalyses from the ECMWF, the current version can use MERRA (Modern-Era Retrospective analysis for Research and Applications) from NASA and JRA-55 (the Japanese 55-year Reanalysis) from the Japanese Meteorological Agency (JMA). The parallelization uses the OMP version of PGI. All arrays are allocated dynamically. The code can be obtained from https://github.com/bernard-legras/traczilla (last access: 30 October 2019).

Author contributions. IP coordinated the contributions to the paper and the code development since version 9, including I/O, updates to turbulent mixing, the implementation of the tests and the distributed version control. ES developed and wrote the description of the parallelized version of FLEXPART and led the assembling of the new code developments into the main model version 10.4. HG developed and tested the new wet deposition scheme for aerosols. NIK contributed to the new wet deposition scheme for aerosols by testing the new model version and coordinated the contributions to the first version of the paper. MC developed the optional new turbulence scheme and the NorESM version and contributed to the WRF version. SE developed and wrote the description of the backward deposition, performed the benchmark test case together with IP, and worked on ECMWF data retrieval and testing. DA and DM contributed to the CTBTO developments including the unified executable, the Vtables approach and testing environment. RLT developed the temporal variation and temperature dependence of the OH reaction. CDGZ developed the dust mobilization scheme around FLEXPART and performed testing of the new model version. NE tested the new model version for black carbon and radionuclide applications. HS implemented the namelist input file format and contributed to the implementation of the NetCDF output and GRIB input routines. LH developed versions 2.0–7.02 of the `flex_extract` retrieval software, in particular the codes for calculating the hybrid coordinate vertical velocity. He also supervised the most recent developments and wrote the description together with AP. SH and DB contributed to the implementation of the NetCDF output module. JB coordinated the development up to FLEXPART version 8.3 and contributed to post-processing the Python module. AF developed the new Phyton-based ECMWF data retrieval software. JB led the development of the WRF and AROME versions and contributed to the turbulence scheme. AP developed, maintained and wrote the description of the `flex_extract` retrieval routines and contributed to the testing environment. PS devised the community website http://flexpart.eu, takes care of tickets and the wiki, contributed to various parts of the code development, and contributed to editing the paper. AS developed the first version of the code in 1998 and supervised all recent developments, including the new settling parameterization for aerosols and time-averaged particle output, and worked on the writing and structuring of the paper.

Competing interests. The authors declare that they have no conflict of interest.

Acknowledgements. The work was performed at the Nordic Center of Excellence eSTICC. Pirmin Kaufmann and Martin Schraner (MeteoSwiss) are acknowledged for code reformatting from fixed Fortran 77 to free Fortran 90 format. We thank Mariëlle Mulder for comments on an early version of this paper. The resources for the numerical simulations were provided by UNINETT Sigma2 (the National Infrastructure for High Performance Computing and Data Storage in Norway) under projects NN9419K and NS9419K. Input wind fields were provided by the ECMWF.

Financial support. This research has been supported by NordForsk (the Nordic Center of Excellence eSTICC, grant no. 57001), the European Research Council (project COMTESSA, grant no. 670462), and the CTBTO (Enhancements of the FLEXPART software on a call-off basis).

Review statement. This paper was edited by Slimane Bekki and reviewed by two anonymous referees.

References

Arnalds, O., Dagsson-Waldhauserova, P., and Olafsson, H.: The Icelandic volcanic aeolian environment: Processes and impacts – A review, Aeolian Res., 20, 176–195, https://doi.org/10.1016/j.aeolia.2016.01.004, 2016.

Arnold, D., Maurer, C., Wotawa, G., Draxler, R., Saito, K., and Seibert, P.: Influence of the meteorological input on the atmospheric transport modelling with FLEXPART of radionuclides from the Fukushima Daiichi nuclear accident, J. Environ. Radioactiv., 139, 212–225, https://doi.org/10.1016/j.jenvrad.2014.02.013, 2015.

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

Atkinson, R.: Gas-phase tropospheric chemistry of volatile organic compounds: 1. Alkanes and alkenes, J. Phys. Chem. Ref. Data, 26, 215–290, 1997.

Balluch, M., and Haynes, P.: Quantification of lower stratospheric mixing processes using aircraft data, J. Geophys. Res., 102, 23487–23504, 1997.

Bey, I., Jacob, D. J., Logan, J. A., and Yantosca, R. M.: Asian chemical outflow to the Pacific in spring: Origins, pathways, and budgets, J. Geophys. Res., 106, 23073–23095, https://doi.org/10.1029/2001jd000806, 2001.

Brioude, J., Arnold, D., Stohl, A., Cassiani, M., Morton, D., Seibert, P., Angevine, W., Evan, S., Dingwell, A., Fast, J. D., Easter, R. C., Pisso, I., Burkhart, J., and Wotawa, G.: The Lagrangian particle dispersion model FLEXPART-WRF version 3.1, Geosci. Model Dev., 6, 1889–1904, https://doi.org/10.5194/gmd-6-1889-2013, 2013.

Bullard, J. E., Baddock, M., Bradwell, T., Crusius, J., Darlington, E., Gaiero, D., Gasso, S., Gisladottir, G., Hodgkins, R., McCulloch, R., Neuman, C. M., Mockford, T., Stewart, H., and Thorsteinsson, T.: High latitude dust in the earth system, Rev. Geophys., 54, 447–485, https://doi.org/10.1002/2016RG000518, 2016.

Cassiani, M., Stohl, A., and Brioude, J.: Lagrangian stochastic modelling of dispersion in the convective boundary layer with skewed turbulence conditions and a vertical density gradient: Formulation and implementation in the FLEXPART Model, Bound.-Lay. Meteorol., 154, 367–390, https://doi.org/10.1007/s10546-014-9976-5, 2015.

Cassiani, M., Stohl, A., Olivié, D., Seland, Ø., Bethke, I., Pisso, I., and Iversen, T.: The offline Lagrangian particle model FLEXPART–NorESM/CAM (v1): model description and comparisons with the online NorESM transport scheme and with

the reference FLEXPART model, Geosci. Model Dev., 9, 4029–4048, https://doi.org/10.5194/gmd-9-4029-2016, 2016.

Cozic, J., Verheggen, B., Mertes, S., Connolly, P., Bower, K., Petzold, A., Baltensperger, U., and Weingartner, E.: Scavenging of black carbon in mixed phase clouds at the high alpine site Jungfraujoch, Atmos. Chem. Phys., 7, 1797–1807, https://doi.org/10.5194/acp-7-1797-2007, 2007.

Dee, D. P., Uppala, S. M., Simmons, A. J., Berrisford, P., Poli, P., Kobayashi, S., Andrae, U., Balmaseda, M. A., Balsamo, G., Bauer, P., Bechtold, P., Beljaars, A. C. M., van de Berg, L., Bidlot, J., Bormann, N., Delsol, C., Dragani, R., Fuentes, M., Geer, A. J., Haimberger, L., Healy, S. B., Hersbach, H., Hólm, E. V., Isaksen, L., Kallberg, P., Köhler, M., Matricardi, M., McNally, A. P., Monge-Sanz, B. M., Morcrette, J.-J., Park, B.-K., Peubey, C., de Rosnay, P., and Tavolato, C., Thépaut, J.-N., and Vitart, F.: The ERA-Interim reanalysis: configuration and performance of the data assimilation system. Q. J. Roy. Meteor. Soc., 137, 553–597, https://doi.org/10.1002/qj.828, 2011.

Eckhardt, S., Prata, A. J., Seibert, P., Stebel, K., and Stohl, A.: Estimation of the vertical profile of sulfur dioxide injection into the atmosphere by a volcanic eruption using satellite column measurements and inverse transport modeling, Atmos. Chem. Phys., 8, 3881–3897, https://doi.org/10.5194/acp-8-3881-2008, 2008.

Eckhardt, S., Cassiani, M., Evangeliou, N., Sollum, E., Pisso, I., and Stohl, A.: Source–receptor matrix calculation for deposited mass with the Lagrangian particle dispersion model FLEXPART v10.2 in backward mode, Geosci. Model Dev., 10, 4605–4618, https://doi.org/10.5194/gmd-10-4605-2017, 2017.

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

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

Fang, X., Shao, M., Stohl, A., Zhang, Q., Zheng, J., Guo, H., Wang, C., Wang, M., Ou, J., Thompson, R. L., and Prinn, R. G.: Top-down estimates of benzene and toluene emissions in the Pearl River Delta and Hong Kong, China, Atmos. Chem. Phys., 16, 3369–3382, https://doi.org/10.5194/acp-16-3369-2016, 2016.

Flesch, T. K., Wilson, J. D., and Lee, E.: Backward-time Lagrangian stochastic dispersion models and their application to estimate gaseous emissions, J. Appl. Meteorol., 34, 1320–1333, 1995.

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., and Stohl, A.: Transport of boreal forest fire emissions from Canada to Europe, J. Geophys. Res., 106, 22887–22906, 2001.

Forster, C., Stohl, A., and Seibert, P.,: Parameterization of Convective Transport in a Lagrangian Particle Dispersion Model and Its Evaluation, J. Appl. Meteorol. Clim., 46, 403–422, https://doi.org/10.1175/JAM2470.1, 2007.

Global Soil Data Task: Global soil data products CD-ROM contents (IGBP-DIS), Data Set, Oak Ridge Natl. Lab. Distrib. Active Arch. Cent., Oak Ridge, Tenn., https://doi.org/10.3334/ORNLDAAC/565, 2014.

Groot Zwaaftink, C. D., Grythe, H., Skov, H., and Stohl, A.: Substantial contribution of northern high-latitude sources to mineral dust in the Arctic, J. Geophys. Res.-Atmos., 121, 13678–13697, https://doi.org/10.1002/2016JD025482, 2016.

Groot Zwaaftink, C. D., Arnalds, Ó., Dagsson-Waldhauserova, P., Eckhardt, S., Prospero, J. M., and Stohl, A.: Temporal and spatial variability of Icelandic dust emissions and atmospheric transport, Atmos. Chem. Phys., 17, 10865–10878, https://doi.org/10.5194/acp-17-10865-2017, 2017.

Groot Zwaaftink, C. D., Henne, S., Thompson, R. L., Dlugokencky, E. J., Machida, T., Paris, J.-D., Sasakawa, M., Segers, A., Sweeney, C., and Stohl, A.: Three-dimensional methane distribution simulated with FLEXPART 8-CTM-1.1 constrained with observation data, Geosci. Model Dev., 11, 4469–4487, https://doi.org/10.5194/gmd-11-4469-2018, 2018.

Grythe, H., Kristiansen, N. I., Groot Zwaaftink, C. D., Eckhardt, S., Ström, J., Tunved, P., Krejci, R., and Stohl, A.: A new aerosol wet removal scheme for the Lagrangian particle model FLEXPART v10, Geosci. Model Dev., 10, 1447–1466, https://doi.org/10.5194/gmd-10-1447-2017, 2017.

Haynes, P. and Anglade, J.: The Vertical-Scale Cascade in Atmospheric Tracers due to Large-Scale Differential Advection, J. Atmos. Sci., 54, 1121–1136, 1997.

Heinz, S.: Statistical Mechanics of Turbulent Flows, Springer, Berlin, Heidelberg, Germany, 214 pp., 2003.

Henne, S., Brunner, D., Oney, B., Leuenberger, M., Eugster, W., Bamberger, I., Meinhardt, F., Steinbacher, M., and Emmenegger, L.: Validation of the Swiss methane emission inventory by atmospheric observations and inverse modelling, Atmos. Chem. Phys., 16, 3683–3710, https://doi.org/10.5194/acp-16-3683-2016, 2016.

Henning, S., Bojinski, S., Diehl, K., Ghan, S., Nyeki, S., Weingartner, E., Wurzler, S., and Baltensperger, U.: Aerosol partitioning in natural mixed-phase clouds, Geophys. Res. Lett., 31, L06101, https://doi.org/10.1029/2003GL019025, 2004.

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

Hittmeir, S., Philipp, A., and Seibert, P.: A conservative reconstruction scheme for the interpolation of extensive quantities in the Lagrangian particle dispersion model FLEXPART, Geosci. Model Dev., 11, 2503–2523, https://doi.org/10.5194/gmd-11-2503-2018, 2018.

Hoinka, K: The tropopause: discovery, definition and demarcation, Meteorol. Z., 6, 281–303, 1997.

Hoyer, S. and Hamman, J.: xarray: N-D labeled Arrays and Datasets in Python, Journal of Open Research Software, 5, p. 10, https://doi.org/http://doi.org/10.5334/jors.148, 2017.

Jones, A., Thomson, D., Hort, M., and Devenish, B.: The UK Met Office's next generation atmospheric dispersion model, NAME III, in Air Pollution Modelling and its Application XVII, edited by: Borrego, C. and Norman, A.-L., Springer, New York, 580–589, 2007.

Kok, J. F.: A scaling theory for the size distribution of emitted dust aerosols suggests climate models underestimate the size of the global dust cycle, P. Natl. Acad. Sci. USA, 108, 1016–1021, https://doi.org/10.1073/pnas.1014798108, 2011.

Kristiansen, N. I., Stohl, A., Prata, A. J., Richter, A., Eckhardt, S., Seibert, P., Hoffmann, A., Ritter, C., Bitar, L., Duck, T. J., and Stebel, K.: Remote sensing and inverse transport modelling of the Kasatochi eruption sulphur dioxide cloud, J. Geophys. Res., 115, D00L16, https://doi.org/10.1029/2009JD013286, 2010.

Kristiansen, N. I., Stohl, A., Prata, F., Bukowiecki, N., Dacre, H., Eckhardt, S., Henne, S., Hort, M., Johnson, B., Marenco, F., Neininger, B., Reitebuch, O., Seibert, P., Thomson, D., Webster, H., and Weinzierl, B.: Performance assessment of a volcanic ash transport model mini-ensemble used for inverse modelling of the 2010 Eyjafjallajökull eruption, J. Geophys. Res., 117, D00U11, https://doi.org/10.1029/2011JD016844, 2012.

Kristiansen, N. I., Prata, A. J., Stohl, A., and Carn, S. A. : Stratospheric volcanic ash emissions from the 13 February 2014 Kelut eruption, Geophys. Res. Lett., 42, 588–596, https://doi.org/10.1002/2014GL062307, 2015.

Kristiansen, N. I., Stohl, A., Olivié, D. J. L., Croft, B., Søvde, O. A., Klein, H., Christoudias, T., Kunkel, D., Leadbetter, S. J., Lee, Y. H., Zhang, K., Tsigaridis, K., Bergman, T., Evangeliou, N., Wang, H., Ma, P.-L., Easter, R. C., Rasch, P. J., Liu, X., Pitari, G., Di Genova, G., Zhao, S. Y., Balkanski, Y., Bauer, S. E., Faluvegi, G. S., Kokkola, H., Martin, R. V., Pierce, J. R., Schulz, M., Shindell, D., Tost, H., and Zhang, H.: Evaluation of observed and modelled aerosol lifetimes using radioactive tracers of opportunity and an ensemble of 19 global models, Atmos. Chem. Phys., 16, 3525–3561, https://doi.org/10.5194/acp-16-3525-2016, 2016.

Kyrö, E.-M., Grönholm, T., Vuollekoski, H., Virkkula, A., Kulmala, M., and Laakso, L. : Snow scavenging of ultrafine particles: field measurements and parameterization. Boreal Environ. Res., 14, 527–538, 2009.

Laakso, L., Grönholm, T., Rannika, Ü., Kosmalea, M., Fiedlera, V., Vehkamäki, H., and Kulmala, M.: Ultrafine particle scavenging coefficients calculated from 6 years field measurements, Atmos. Environ. 37, 3605–3613, https://doi.org/10.1016/S1352-2310(03)00326-1, 2003.

Laloyaux, P., de Boisseson, E., Balmaseda, M., Bidlot, J.R. Broennimann, S., Buizza, R., Dalhgren, P., Dee, D., Haimberger, L., Hersbach, H., Kosaka, Y., Martin, M., Poli, P., Rayner, N., Rustemeier, E., and Schepers, D.: CERA20C: A coupled reanalysis of the twentieth century, J. Adv. Model Earth Sy., 10, 1172–1195, https://doi.org/10.1029/2018MS001273, 2018.

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

Lin, J. C., Gerbig, C., Wofsy, S. C., Andrews, A. E., Daube, B. C., Davis, K. J., and Grainger, C. A.: A near-field tool for simulating the upstream influence of atmospheric observations: The Stochastic Time-Inverted Lagrangian Transport (STILT) model, J. Geophys. Res., 108, 4493, https://doi.org/10.1029/2002JD003161, 2003.

Luhar, A. K. and Britter, R. E.: Random walk model for dispersion in inhomogeneous turbulence in a convective boundary layer, Atmos. Environ., 23, 1911–1924, 1989.

Marticorena, B. and Bergametti, G.: Modeling the atmospheric dust cycle: 1. Design of a soil-derived dust emission scheme, J. Geophys. Res., 100, 16415–16430, https://doi.org/10.1029/95JD00690, 1995.

McConnell, J. R., Wilson, A. I., Stohl, A., Arienzo, M. M., Chellman, N. J., Eckhardt, S., Thompson, E. M., Pollard, A. M., and Steffensen, J. P.: Lead pollution recorded in Greenland ice indicates European emissions tracked plagues, wars, and imperial expansion during antiquity, P. Natl. Acad. Sci. USA, 115, 5726–5731, https://doi.org/10.1073/pnas.1721818115, 2018.

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

Moxnes, E., Kristiansen, N.I., Stohl, A., Clarisse, L., Durant, A., Weber, K., and Vogel, A.: Separation of ash and sulfur dioxide during the 2011 Grímsvötn eruption, J. Geophys. Res.-Atmos., 119, 7477–7501, https://doi.org/10.1002/2013JD021129, 2014.

Message Passing Interface Forum: available at: https://www.mpi-forum.org/ (last access: 30 October 2019), 2015.

Naeslund, E. and Thaning, L.: On the settling velocity in a nonstationary atmosphere. Aerosol Sci. Tech., 14, 247–256, 1991.

Oney, B., Henne, S., Gruber, N., Leuenberger, M., Bamberger, I., Eugster, W., and Brunner, D.: The CarboCount CH sites: characterization of a dense greenhouse gas observation network, Atmos. Chem. Phys., 15, 11147–11164, https://doi.org/10.5194/acp-15-11147-2015, 2015.

Ottino, J.: The Kinematics of Mixing: Stretching, Chaos and Transport, Cambridge Univ. Press, New York, 1989.

Pisso, I. and Legras, B.: Turbulent vertical diffusivity in the sub-tropical stratosphere, Atmos. Chem. Phys., 8, 697–707, https://doi.org/10.5194/acp-8-697-2008, 2008.

Pisso, I., Real, E., Law, K. S., Legras, B., Bousserez, N., Attié, J. L., and Schlager, H.: Estimation of mixing in the troposphere from Lagrangian trace gas reconstructions during long-range pollution plume transport, J. Geophys. Res., 114, D19301, https://doi.org/10.1029/2008JD011289, 2009.

Pisso, I., Sollum, E., Grythe, H., Kristiansen, N.I., Cassiani, M., Eckhardt, S., Arnold, D., Morton, D., Thompson, R.L., Groot Zwaaftink, C.D., Evangeliou, N., Sodemann, H., Haimberger, L., Henne, S., Brunner, D., Burkhart, J.F., Fouilloux, A., Brioude, J., Philipp, A., Seibert, P., and Stohl, A.: FLEXPART 10.4 (Version 10.4), Geosci. Model Dev. Discuss. Zenodo, https://doi.org/10.5281/zenodo.3542278, 2019.

Ramli, Huda Mohd. and Esler, J. G.: Quantitative evaluation of numerical integration schemes for Lagrangian particle dispersion models, Geosci. Model Dev., 9, 2441–2457, https://doi.org/10.5194/gmd-9-2441-2016, 2016.

Rastigejev, Y., Park, R., Brenner, M., and Jacob, D.: Resolving intercontinental pollution plumes in global models of atmospheric transport, J. Geophys. Res., 115, D02302, https://doi.org/10.1029/2009JD012568, 2010.

Reithmeier, C. and Sausen, R.: ATTILA: Atmospheric Tracer Transport in a Lagrangian Model, Tellus B, 54, 278–299, 2002.

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

Seibert, P. and Frank, A.: Source-receptor matrix calculation with a Lagrangian particle dispersion model in backward mode, Atmos. Chem. Phys., 4, 51–63, https://doi.org/10.5194/acp-4-51-2004, 2004.

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

Shao, Y. and Lu, H.: A simple expression for wind erosion threshold friction velocity, J. Geophys. Res., 105, 22437–22443, https://doi.org/10.1029/2000JD900304, 2000.

Sodemann, H., Lai, T. M., Marenco, F., Ryder, C. L., Flamant, C., Knippertz, P., Rosenberg, P., Bart, M., and McQuaid, J. B.: Lagrangian dust model simulations for a case of moist convective dust emission and transport in the western Sahara region during Fennec/LADUNEX, J. Geophys. Res.-Atmos., 120, 6117–6144, https://doi.org/10.1002/2015JD023283, 2015.

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

Stein, A. F., Draxler, R. R, Rolph, G. D., Stunder, B. J. B., Cohen, M. D., and Ngan, F.: NOAA's HYSPLIT atmospheric transport and dispersion modeling system, B. Am. Meteorol. Soc., 96, 2059–2077, https://doi.org/10.1175/BAMS-D-14-00110.1, 2015.

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

Stohl, A., and James, P.: A Lagrangian analysis of the atmospheric branch of the global water cycle: 1. Method description, validation, and demonstration for the August 2002 flooding in Central Europe, J. Hydrometeorol., 5, 656–678, 2004.

Stohl, A. and Thomson, D. J.: A density correction for Lagrangian particle dispersion models, Bound.-Lay. Meteorol., 90, 155–167, 1999.

Stohl, A. and Trickl, T.: 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, 30445–30462, 1999.

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

Stohl, A., Hittenberger, M., and Wotawa, G.: Validation of the Lagrangian particle dispersion model FLEXPART against large scale tracer experiment data, Atmos. Environ., 32, 4245–4264, 1998.

Stohl, A., Haimberger, L., Scheele, M. P., and Wernli, H.: An intercomparison of results from three trajectory models, Meteorol. Appl., 8, 127–135, 2001.

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

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

Stohl, A., Cooper, O. R., Damoah, R., Fehsenfeld, F. C., Forster, C., Hsie, E.-Y., Hübler, G., Parrish, D. D., and Trainer, M.: Forecasting for a Lagrangian aircraft campaign, Atmos. Chem. Phys., 4, 1113–1124, https://doi.org/10.5194/acp-4-1113-2004, 2004.

Stohl, A., Forster, C., Frank, A., Seibert, P., and Wotawa, G.: Technical note: The Lagrangian particle dispersion model FLEXPART version 6.2, Atmos. Chem. Phys., 5, 2461–2474, https://doi.org/10.5194/acp-5-2461-2005, 2005.

Stohl, A., Seibert, P., Arduini, J., Eckhardt, S., Fraser, P., Greally, B. R., Lunder, C., Maione, M., Mühle, J., O'Doherty, S., Prinn, R. G., Reimann, S., Saito, T., Schmidbauer, N., Simmonds, P. G., Vollmer, M. K., Weiss, R. F., and Yokouchi, Y.: An analytical inversion method for determining regional and global emissions of greenhouse gases: Sensitivity studies and application to halocarbons, Atmos. Chem. Phys., 9, 1597–1620, https://doi.org/10.5194/acp-9-1597-2009, 2009.

Stohl, A., Prata, A. J., Eckhardt, S., Clarisse, L., Durant, A., Henne, S., Kristiansen, N. I., Minikin, A., Schumann, U., Seibert, P., Stebel, K., Thomas, H. E., Thorsteinsson, T., Tørseth, K., and Weinzierl, B.: Determination of time- and height-resolved volcanic ash emissions and their use for quantitative ash dispersion modeling: the 2010 Eyjafjallajökull eruption, Atmos. Chem. Phys., 11, 4333–4351, https://doi.org/10.5194/acp-11-4333-2011, 2011.

Stohl, A., Seibert, P., Wotawa, G., Arnold, D., Burkhart, J. F., Eckhardt, S., Tapia, C., Vargas, A., and Yasunari, T. J.: Xenon-133 and caesium-137 releases into the atmosphere from the Fukushima Dai-ichi nuclear power plant: determination of the source term, atmospheric dispersion, and deposition, Atmos. Chem. Phys., 12, 2313–2343, https://doi.org/10.5194/acp-12-2313-2012, 2012.

Stohl, A., Klimont, Z., Eckhardt, S., Kupiainen, K., Shevchenko, V. P., Kopeikin, V. M., and Novigatsky, A. N.: Black carbon in the Arctic: the underestimated role of gas flaring and residential combustion emissions, Atmos. Chem. Phys., 13, 8833–8855, https://doi.org/10.5194/acp-13-8833-2013, 2013.

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

Sutherland, W.: The viscosity of gases and molecular force, Philos. Mag., 36, 507–531, 1893.

Tateishi, R., Hoan, N. T., Kobayashi, T., Alsaaideh, B., Tana, G., and Phong, D. X.: Production of Global Land Cover Data-GLCNMO2008, J. Geogr. Geol., 6, 99–122, https://doi.org/10.5539/jgg.v6n3p99, 2014.

Tegen, I. and Fung, I.: Modeling of mineral dust in the atmosphere: Sources, transport, and optical thickness, J. Geophys. Res., 99, 22897–22914, https://doi.org/10.1029/94JD01928, 1994.

Thompson, R. L. and Stohl, A.: FLEXINVERT: an atmospheric Bayesian inversion framework for determining surface fluxes of trace species using an optimized grid, Geosci. Model Dev., 7, 2223–2242, https://doi.org/10.5194/gmd-7-2223-2014, 2014.

Thompson, R. L., Stohl, A., Zhou, L. X., Dlugokencky, E., Fukuyama, Y., Tohjima, Y., Kim, S.-Y., Lee, H., Nisbet, E. G., Fisher, R. E., Lowry, D., Zhao, G., Weiss, R. F., Prinn, R. G., O'Doherty, S., Fraser, P., and White, J. W. C.: Methane emissions in East Asia for 2000–2011 estimated using an atmospheric Bayesian inversion, J. Geophys. Res., 120, 4352–4369, https://doi.org/10.1002/2014JD022394, 2015.

Thompson, R. L., Sasakawa, M., Machida, T., Aalto, T., Worthy, D., Lavric, J. V., Lund Myhre, C., and Stohl, A.: Methane fluxes in the high northern latitudes for 2005–2013 estimated using a Bayesian atmospheric inversion, Atmos. Chem. Phys., 17, 3553–3572, https://doi.org/10.5194/acp-17-3553-2017, 2017.

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

Thomson, D. J.: A stochastic model for the motion of particle pairs in isotropic high-Reynolds-number turbulence, and its application to the problem of concentration variance, J. Fluid Mech., 210, 113–153, 1990.

Thomson, D. J. and Wilson, J. D.: History of Lagrangian stochastic models for turbulent dispersion, in: Lagrangian modelling of the atmosphere, edited by: Lin, J., Brunner, D., Gerbig, C., Stohl, A., Luhar, A., and Webleyv, P., American Geophysical Union, Washington, DC, 2013.

Tiedtke, M.: Representation of clouds in large-scale models, Mon. Weather Rev., 121, 3040–3061, 1993.

Uliasz, M.: Lagrangian particle dispersion modeling in mesoscale applications, in: Environmental Modeling, Vol. II, edited by: Zannetti, P., Computational Mechanics Publications, Southampton, UK, 1994.

Venkatram, A. and Wyngaard, J. C. (Eds.): Lectures on Air Pollution Modeling, American Meteorological Society, ISBN 9780933876675, 390 pp., Boston, 1988.

Verreyken, B., Brioude, J., and Evan, S.: Development of turbulent scheme in the FLEXPART-AROME v1.2.1 Lagrangian particle dispersion model, Geosci. Model Dev., 12, 4245–4259, https://doi.org/10.5194/gmd-12-4245-2019, 2019a.

Verreyken, B., Brioude, J., and Evan, S.: Development of turbulent scheme in the FLEXPART-AROME v1.2.1 Lagrangian particle dispersion model, Geosci. Model Dev. Discuss., https://doi.org/10.5194/gmd-2019-89, in review, 2019b.

Weil, J. C.: Updating Applied Diffusion Models, Lectures on Air Pollution Modeling, edited by: Venkatram, A. and Wyngaard, J. C., J. Appl. Meteorol. Clim., 24, 1111–1130, 1985.

Wilson, J. D. and Sawford, B. L.: Review of Lagrangian stochastic models for trajectories in the turbulent atmosphere, Bound.-Lay. Meteorol., 78, 191–210, 1996.

Wilson, J. D., Thurtell, G. W., and Kidd, G. E.: Numerical simulation of particle trajectories in inhomogeneous turbulence part II: Systems with variable turbulence velocity scale, Bound.-Lay. Meteorol., 21, 423–441, 1981.

Wotawa, G., DeGeer, L.-E., Denier, P., Kalinowski, M., Toivonen, H., D'Amours, R., Desiato, F., Issartel, J.-P., Langer, M., Seibert, P., Frank, A., Sloan, C., and Yamazawa, H.:, Atmospheric transport modelling in support of CTBT verification – overview and basic concepts, Atmos. Environ., 37, 2529–2537, 2003.

Zannetti, P.: Particle Modeling and Its Application for Simulating Air Pollution Phenomena, in: Environmental Modelling, edited by: Melli, P. and Zannetti, P., Computational Mechanics Publications, Southampton, UK, 1992.