

Met Office

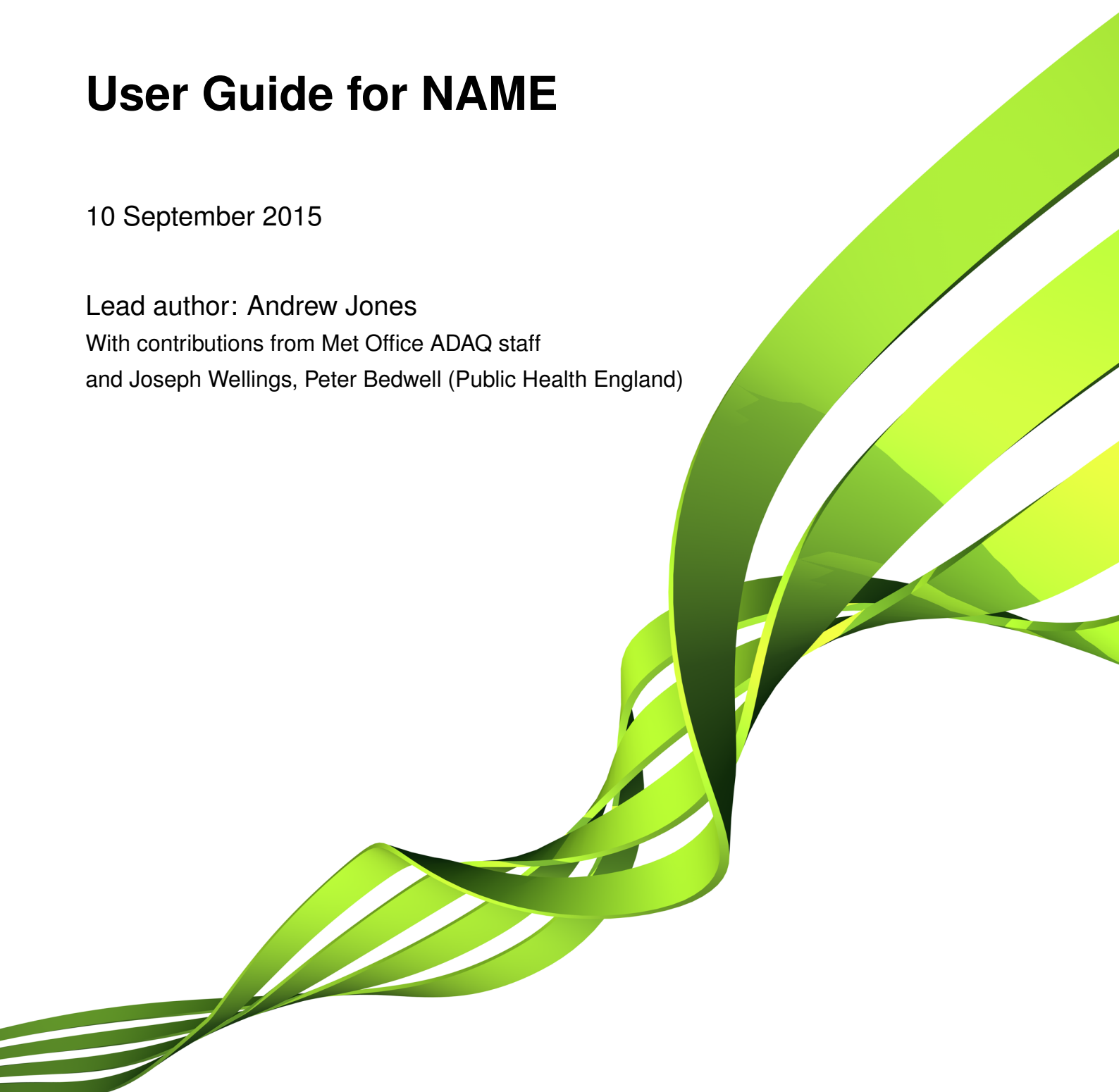
User Guide for NAME

10 September 2015

Lead author: Andrew Jones

With contributions from Met Office ADAQ staff

and Joseph Wellings, Peter Bedwell (Public Health England)



User Guide for NAME

Lead author: Andrew Jones

Preface

Preparing this user guide represents the first significant effort at writing a comprehensive description of NAME since TDN 262 in 1999. Although that TDN still reflects many aspects of the underpinning science in NAME, many other aspects of the model have changed considerably over the intervening years. New features and capabilities have also been introduced far beyond those of the earlier modelling system. It is therefore entirely appropriate that a new user guide should now be written.

The *User Guide for NAME* complements the existing user documentation available for NAME, and is intended to be read along side those other documents. The author wishes to acknowledge the contribution to NAME documentation that has been made over many years by past and present members of the Met Office's Atmospheric Dispersion Group. Some of this material is included directly within this user guide, but the vast bulk of existing NAME documentation is accessed in the guide through a comprehensive set of references.

Andrew Jones, Met Office – October 2013

Update to Version 2.0

The second edition of a *User Guide for NAME* has been prepared for use with NAME version 6.4. This is the first version of NAME to incorporate a fully functional Eulerian submodel representing advection, vertical diffusion, deposition and sedimentation processes. The chemistry functionality within NAME has been migrated into this new Eulerian modelling framework, and the chemistry interface redesigned to be more flexible and allow the introduction of alternative chemistry schemes. A new 'deep' convective mixing scheme, based on a mass-flux approach, has been developed for representing vertical transport in convective clouds, and the radar met functionality has been completely redesigned to give greater flexibility and allow use of radar-derived rainfall estimates from the Met Office's UKPP and EuroPP gridded post-processing systems to improve the modelling of deposition 'hot-spots'. A further enhancement is the use of a revised and more accurate calculation of particle trajectories in latitude-longitude coordinate systems, which will reduce trajectory errors especially near to the poles.

Andrew Jones, Met Office – December 2014

Update to Version 3.0

The third edition of a *User Guide for NAME* has been prepared for use with NAME version 6.5. This version of NAME includes further enhancements to the Eulerian modelling framework, with radioactive decay and agent decay processes now represented, and Eulerian deposition outputs available to the user. Building on the more flexible framework for chemistry introduced in version 6.4, an alternative chemistry scheme has been added to model the conversion between gaseous and particulate forms of radioactive iodine. Alternative sedimentation schemes (or ‘particle shape’ schemes) have been introduced for the fall velocity of non-spherical volcanic ash particles, and a met-dependent iterative plume-rise model has been developed to estimate mass release given an observed plume-rise height and the prevailing meteorology. A met-dependent parametrization of unresolved mesoscale motions (previously referred to as ‘meander’) is now available, in which the scheme parameters (velocity variances and Lagrangian timescales) are dependent on the input meteorological data and are specified as part of the met definitions. Support for NWP met data is further enhanced in this version of NAME, which includes an ability to read ‘packed’ PP files for UM met data and also to read NetCDF met files from other NWP models. Finally it is now possible for the user to request particle size-segregated output quantities and, partly to accommodate this, a new file format (currently experimental) has been developed for fields output from NAME.

Andrew Jones, Met Office – September 2015

Copyright notice

The NAME software, documentation and supporting materials are owned by the Met Office and are legally protected under Crown Copyright. Use of NAME is strictly restricted to registered users holding a valid licence agreement with the Met Office (see Section 3.3 for further details about licensing of NAME).

Contents

Preface	1
Copyright notice	2
Contents	3
Conventions used in this guide	6
Introduction	7
1 NAME – Model Description	11
1.1 Model Overview	11
1.2 Core Model Infrastructure	14
1.2.1 Physical Constants and Units	14
1.2.2 Times	14
1.2.3 Arrays	16
1.2.4 Particle size distributions	16
1.2.5 Coordinate Systems	18
1.2.6 Locations, Grids and Domains	20
1.2.7 Advanced modelling options and controls	26
1.3 NAME Inputs	35
1.3.1 Filename conventions for NAME	35
1.3.2 Command line arguments	35
1.3.3 ‘Headed’ input files for NAME	35
1.3.4 Other files supplying input data for NAME	36
1.4 NAME Outputs	37
1.4.1 Requesting output of fields from NAME	37
1.4.2 Requesting trajectory-based output (particle/puff information) from NAME	49
1.4.3 Requesting output of pdfs for concentration fluctuations	50
1.4.4 Format of NAME output files	51
1.5 Source Terms in NAME	59
1.5.1 Defining species information in NAME	59
1.5.2 Defining species use information in NAME	63
1.5.3 Defining cloud gamma parameters for radiological modelling	63

1.5.4	Defining source terms in NAME	64
1.5.5	Defining source groups in NAME	72
1.5.6	Controlling particle numbers in a NAME run	73
1.6	Meteorology in NAME	75
1.6.1	Using observations from a weather station (single-site meteorology)	76
1.6.2	Using meteorological fields from a Numerical Weather Prediction model	78
1.6.3	Using rainfall estimates from weather radar	89
1.6.4	Advanced options for flow information	92
1.7	Physical Processes in NAME	94
1.7.1	Specifying dispersion modelling options	94
1.7.2	Running NAME with particles	94
1.7.3	Running NAME with puffs	97
1.7.4	Advection and diffusion	100
1.7.5	Vertical mixing by deep convective clouds	109
1.7.6	Plume rise scheme for buoyant and momentum-driven releases	110
1.7.7	Deposition and sedimentation processes	111
1.7.8	Radiological modelling	121
1.7.9	Other decay processes	126
1.7.10	NAME chemistry modelling	127
1.7.11	Eulerian model	132
1.7.12	Short-term concentration fluctuations	134
1.8	Code Management, Compilation and Optimisation	137
1.8.1	Code structure and programming standards	137
1.8.2	Compiling NAME	137
1.8.3	Running NAME	140
1.8.4	Parallelisation with OpenMP	141
1.8.5	Profiling and the use of timers	144
2	NAME – Supplementary Utilities	147
2.1	Meteorological Utilities for NAME	147
2.1.1	Producing ADMS-style meteorology files with NAME	148
2.1.2	Reading GRIB format meteorological data	149
2.1.3	Reading NetCDF format meteorological data	152
2.2	Producing Graphics from NAME Output	154
2.2.1	Generating graphics using IDL	154
2.2.2	Generating graphics using Python	156
2.3	Post-processing of NAME Output	157
2.3.1	Converting NAME output to GRIB format	157



2.3.2	Converting NAME output to NetCDF format	157
3	NAME – Further Aspects	159
3.1	Documentation	159
3.2	Testing and Validation	162
3.3	NAME Licensing	163
3.4	Installation and Model Upgrades	164
3.5	NAME Training	165
3.6	NAME User Community	166
3.7	NAME Bibliography	167
	Bibliography	175
	Index	180

Conventions used in this guide

The Met Office dispersion model is generally referred to as '**NAME**' within this user guide, although for historical reasons the phrase '**NAME III**' has sometimes appeared in other literature to indicate the most recent form of the modelling system (and to explicitly distinguish it from its predecessor '**NAME II**'). The '**NAME II**' model has been formally retired for several years and so today there is less risk of confusion between the 'old' and 'new' modelling systems. Consequently the labels '**NAME**' and '**NAME III**' are generally used interchangeably, with a reference to '**NAME**' implicitly meaning the '**NAME III**' system. However '**NAME II**' is still mentioned on a few occasions within the text in connection with a deprecated output file format (referred to as NAME II format) which is supported by '**NAME III**' for legacy systems.

It is also acknowledged that the terminology surrounding words such as *particle*, *particulate* and *aerosol* might sometimes be confusing. All three of these words can refer to both sedimenting and non-sedimenting particles, though effort has been made to clearly distinguish between sedimenting and non-sedimenting material where this is important. For clarity, the term *particle* will mainly be used in this user guide in preference to *particulate* or *aerosol*.

The following conventions and syntax are adopted within this user guide.

- Standard text appears in plain font with occasional use of *italic* or **bold** styles where emphasis is intended.
- Highlighting is used for **command-line arguments** or **options** called when invoking NAME.
- Highlighting is used for **Filenames** and **\Folder\Paths**.
- References to names of input blocks appear as **Block Name**.
- References to keyword variable names within input blocks appear as **Keyword Name**.
- References to keyword variable values within input blocks appear as **Keyword Value**.
- References to other sections within this user guide appear as **SECTION NAME**.
- References to external documentation appear as **>> EXTERNAL DOCUMENT <<**.
- The  symbol is used to highlight an area where additional caution is required by the user. These warnings cover aspects of the model set up that are more difficult, or perhaps less intuitive, and so there is a higher risk of a misunderstanding over what the model is doing or of a user making an error in their set up.
- The  symbol gives helpful 'Hints and Tips' on using NAME!

Introduction

The **Numerical Atmospheric-dispersion Modelling Environment**, or **NAME**, is the Met Office's atmospheric dispersion model. It is a Lagrangian model designed to predict the atmospheric transport and deposition to the ground surface of airborne substances, and treats both gaseous and particulate materials. NAME is used for a wide range of activities that include emergency-response modelling, routine forecasting applications, scientific research and policy support work.

As a Lagrangian model, NAME uses Monte Carlo random-walk techniques to represent within a stochastic framework the turbulent transport of pollutants in the atmosphere. Emissions into the atmosphere from a pollutant source are simulated by creating a large number of model objects called 'particles' (sometimes referred to as 'model particles' to make explicit the distinction between these computational objects and actual physical particles). Each (model) particle is intended to represent a certain proportion of the mass or activity of released pollutants. These particles are then advected along by the ambient three-dimensional wind field of a 'model atmosphere' with turbulent dispersion processes being simulated using random-walk methods. The particles can also evolve with time to account for various atmospheric processes that might transform or remove the pollutants (e.g. radioactive decay, chemical transformations, or dry and wet deposition). Air concentration and deposition fields are calculated by applying suitable spatial (and possibly temporal) averaging to the particle masses or activities at the required output times. There is also a puff scheme available in NAME that aims to represent material spread explicitly, although its applicability is restricted to a limited range of modelling options.

NAME supports various approaches for defining a 'model atmosphere' to use in its simulation. Most commonly, NAME uses one of two options: single-site meteorology, where NAME reads hourly records of basic weather parameters measured at an observation site; or NWP meteorology, where NAME reads gridded meteorological fields from a numerical weather prediction model such as the Met Office's Unified Model. The Met Office have extensive archives of station weather observations and NWP data sets that can be supplied for use with NAME. Advanced NAME options also exist, such as the capability to utilise high resolution rainfall estimates derived from a weather radar network, or the ability to use a coupled linear flow model to enhance the description of a flow field by including a representation of local terrain effects.

The NAME model is currently supported for Windows and Linux operating systems, although it has been successfully compiled under the Sun Solaris and Apple Mac environments, and recently also for the IBM HPC environment (e.g. the Met Office supercomputer). As the underlying source code for NAME is written in standard Fortran 95, the model could, at least in principle, be ported to any platform supported by a suitable compiler. NAME is supplied as a standalone executable file, which can be invoked from the command line or called from within a script (it is also possible to launch NAME under the Windows operating system by simply double clicking on its application icon). Within the Met Office Operations Centre, NAME has also been integrated into a web-based

emergency-response interface designed to support forecaster activities.

All raw NAME output files are in plain text ASCII format, and consist of a file header description followed by the requested output data. These files can be inspected manually using any basic file editor, however NAME output files are normally imported into other applications (e.g. to create graphical products using IDL, python or GIS tools). NAME output files may also be converted to other formats, such as GRIB or netCDF, by downstream processing.

NAME was originally developed as a nuclear accident model in response to the Chernobyl nuclear disaster in 1986, and it continues to have an important operational role within UK and international frameworks for responding to radiological incidents (e.g. RIMNET, RSMC, CTBTO). Over the years, the radiological capabilities of NAME have been further enhanced, including the relatively recent additions of decay-chain modelling and cloud gamma dose calculations. However NAME has also evolved in a much broader sense as a general-purpose atmospheric dispersion model, with developments such as an atmospheric chemistry scheme.

Alongside these scientific and technical developments in NAME, there has come a growing list of applications where NAME has been used to provide emergency-response assessments and advice. Aside from its radiological responsibilities, NAME is also used operationally by the Met Office in its role as the London VAAC (Volcanic Ash Advisory Centre) to model volcanic ash transport following major eruptions in the Icelandic region (such as the Eyjafjallajökull eruption in 2010 and Grimsvotn eruption of 2011). NAME also underpins the Met Office CHEMET service to the UK emergency services, providing area-at-risk assessments in the event of hazardous releases such as chemical spills or large factory fires. Finally, in recent years, NAME modelling results have been increasingly used in support of emergency-response epidemiological studies in the human, plant and animal health communities (e.g. the UK outbreaks of foot-and-mouth disease in 2001 and 2007, bluetongue, Legionnaire's disease or Chalara ash dieback).

Aside from its use in emergency-response, the NAME model also has routine applications such as the production of daily air quality forecasts¹ for the United Kingdom. NAME is used extensively for scientific research studies, especially in conjunction with measurements of atmospheric pollutants. For instance, NAME has been applied to analyse and help understand trace gas measurements at sites such as Mace Head on the western coast of Ireland. Combining NAME results with such observations enables estimates to be deduced, for example, of trends in the baseline background concentrations of radiatively important gases, or of the geographical distribution of emission sources. Equally NAME might also be employed to assist our scientific understanding in support of policy decisions (e.g. in modelling the consequences of emission reduction strategies or the potential impact of future climate scenarios for air quality).

Traditionally, the NAME model has been run 'in-house' at the Met Office, with the NAME output data and/or graphical products being distributed externally as required. More recently, however, we

¹NAME has recently been superseded by the Unified Model for UK air quality forecasting, however the capability still exists and routine products such as trajectories are still produced by NAME

have been aiming to encourage a wider use of NAME for research purposes outside the Met Office. A growing community of external NAME users is gradually developing, which now includes several UK regulatory organisations, as well as universities and other research institutions within the UK and beyond. This initiative is still largely in its infancy, and our aim is for a wider community network to evolve over the coming years.

As with any dispersion model, NAME has its strengths and its limitations. Fundamentally these are related to the essence of the NAME modelling methodology – that NAME is a Lagrangian particle model.

In the Lagrangian approach, pollutants are represented using a *finite number of discrete particles*, which are tracked individually through the flow. This discretised representation of the mass introduces effective limits on the minimum concentrations, etc. that can be adequately resolved in any particular model run. For instance, from a theoretical viewpoint (and excluding time averaging considerations), the smallest non-zero concentration that can be represented would occur for a grid box occupied by a single model particle. This theoretical lower bound on the concentration would equate to the mass carried on the one particle divided by the volume of the grid box under consideration. However, due to the stochastic nature of the dispersion, adequate sampling of particles within a grid box is necessary to provide statistically robust estimates of the concentration (in the case of a single particle in a grid box, for example, a small change in the trajectory of that particle could place it into a different grid box, and so the original presence of a non-zero concentration is partly down to just chance here). Formally, the error in the concentration estimate is expected to behave as the inverse of the square root of the number of particles in the sample. At low particle number densities, concentration estimates are ‘quantised’ into discrete multiples of these basic quanta of concentrations, and large relative changes in concentration can occur as a result (e.g. a doubling in concentration occurs between the situation where there is one particle in a grid box and the case where there are two particles present). The precise concentration values at the level of this statistical ‘noise’ should not be relied upon. Instead, all that can be meaningfully stated is that there is a possibility of some material being present in relatively low concentrations within these areas. Statistical noise is a particular issue in the tails of plumes where particle number densities are low by definition.

When designing a model run configuration for NAME, the requirements for output accuracy (with its implications for the number of particles to release) need to be balanced against other computational demands of the model. Note also that NAME has a puff scheme for use at short ranges, which is intended to overcome the issues raised here concerning particle numbers and statistical noise (although the puff scheme does adopt certain approximations and it is not fully compatible with all other modelling options in NAME).

Beyond the issues around particle noise, the treatment of certain model processes (e.g. chemistry) can be less natural and more difficult to handle within a Lagrangian framework, although such difficulties can be overcome. On the other hand, advanced Lagrangian models such as NAME

provide distinct benefits over other dispersion modelling strategies, which is one reason for their widespread use today for atmospheric dispersion applications. For instance, Lagrangian models can deal quite naturally with localised releases such as point sources (which are difficult to treat in an Eulerian framework, say) and they tend to be very efficient at accurately representing relatively narrow plumes encountered near to point sources or the filamentary structures often observed in plumes at large downwind distances. Lagrangian models are also efficient in representing multiple species (potentially a large number of them) in a simulation.

Acknowledging those situations where an Eulerian approach can provide benefits over the traditional Lagrangian particle approach of NAME, an Eulerian sub-model has been developed within NAME to provide the user with a hybrid Lagrangian-Eulerian scheme. Pollutants are initially released from sources on particles but can later be transferred onto scalar fields for advection by an Eulerian model using a semi-Lagrangian method based on the scheme used in the *ENDGame* dynamical core of the Unified Model. The functionality of the Eulerian model also includes vertical diffusion, sedimentation of sedimenting material, and dry and wet deposition. The Eulerian model is integrated with NAME's chemistry scheme and is well suited to modelling the dispersion of species such as ozone that evolve smoothly over large spatial scales.

NAME also benefits from its association with the Met Office's world-leading Numerical Weather Prediction model, the Unified Model (MetUM), which provides three-dimensional gridded meteorological fields for dispersion modelling ranging from the global scale to kilometre-scale (over the UK).² Moreover, NAME is designed in a flexible and extendible manner, both in terms of its underlying code design and in the specification of any particular model simulation. Flexibility exists in the various options that can be applied as and when required by the user, while at the same time the modelling framework is flexible enough to support potential enhancements to extend its capabilities. For instance, for NAME to use a different source of meteorological data or even to couple NAME with an entirely new off-line flow field scheme such as a CFD (computational fluid dynamics) model.

²In recent years, very high resolution simulations with the MetUM at grid scales of 100 metres or less have also started to be performed, and a recent research study with NAME, for instance, has investigated running a dispersion simulation with input meteorological fields at 150 metre resolution.

Chapter 1

NAME – Model Description

1.1 Model Overview

NAME is a state-of-the-art Lagrangian atmospheric dispersion model. Its main features are summarised in Table 1.1 below. Note that although NAME is primarily Lagrangian in nature, it also uses non-Lagrangian techniques for modelling certain processes (e.g. chemistry).

Model description	
Model Name	NAME (Numerical Atmospheric dispersion Modelling Environment)
Organisation	UK Met Office
Purpose	Atmospheric transport and dispersion model
Modelling Method	Numerical
Computational Approach	Lagrangian (with embedded Eulerian sub-model)
Principal Applications	Emergency response; scientific research; policy support activities
Conditions of Use	Under licence – currently for non-commercial research use only
Basic infrastructure	
Horizontal coordinate systems	Latitude-Longitude (standard or with a rotated pole); UK National Grid; Polar Stereographic projections; Transverse Mercator projections
Vertical coordinate systems	Height (above ground or above sea level); pressure; flight levels in ICAO standard atmosphere; height-based and pressure-based hybrid systems (used by NWP models)
Spatial scales	$\sim 10^2$ metres \rightarrow global
Temporal scales	Minutes \rightarrow years
Areal coverage	Local ($\sim 10^2$ m \rightarrow ~ 10 km)
	Regional (~ 10 km \rightarrow $\sim 10^2$ km)
	Global ($\sim 10^2$ km \rightarrow global)
Temporal coverage	Forecast runs (out to 6 days ahead for global; 36 hours for UK)
	Analysis runs (hours \rightarrow multi-year simulations)
	Re-analysis data sets available back to 1957
	‘Forwards’ and ‘backwards’ run modes

Table 1.1: Overview of NAME characteristics and features

Physical processes	
Advection and diffusion	Three-dimensional random-walk techniques of varying levels of sophistication: diffusive scheme (computationally efficient); Langevin-type scheme (greater accuracy); puff approach for short range applications
Turbulence schemes	Turbulence and unresolved mesoscale motions treated independently within the boundary layer Constant-magnitude free tropospheric turbulence applied above the boundary layer
Dry deposition	General scheme based on surface resistance / deposition velocity Land surface dependent scheme for certain gaseous species
Wet deposition	Rain out ('in-cloud' removal) and wash out ('below-cloud' removal by rain impaction)
Particle sedimentation	Based on Stokes flow with the Cunningham correction applied for small particle sizes Alternative sedimentation schemes available for non-spherical volcanic ash particles
Plume rise	Represents buoyancy and momentum-driven releases Based on conservation equations of mass, momentum and heat (development on Briggs formulae)
Radiological decay	Simple half-life decay of radionuclides; decay chains; cloud gamma dose assessments
Physical decay	Power law decay, decay of biological agents and vector-borne species
Eulerian sub-model	Semi-Lagrangian advection, with treatment of vertical diffusion, sedimentation, radiological decay, agent decay, dry deposition and wet deposition
Chemistry	Comprehensive sulphur/nitrogen/hydrocarbon chemistry scheme based on global atmospheric chemistry model STOCHEM Iodine chemistry scheme modelling gaseous-to-particulate conversion of radioactive iodine
Source terms	
Source geometry	Point, line, area and volume
Source shape	Cuboid, Ellipsoid or Cylindroid Gaussian or Uniform distribution cross-sections
Composite sources	Most practical source configurations can be represented using composites of the above basic source types
Species characteristics	Multiple species can be represented, with separate physical and chemical characteristics for each species

Table 1.1: Overview of NAME characteristics and features (cont.)

Input meteorological data	
Single-site meteorology	Hourly records of basic weather variables (e.g. observations from a weather station)
NWP meteorology	Three-dimensional gridded data parameters from Numerical Weather Prediction models Met Office Unified Model (MetUM): UK (1.5 km) → Global (17 km) ECMWF forecasts (deterministic global and ensemble) ECMWF reanalysis products (ERA-40, ERA-Interim)
Rainfall radar data	UKPP/EuroPP (2 km / 5 km precipitation analyses)
NWP model coupling	Off-line coupling, including ability to 'nest' data at different scales
NWP met file frequency	Typically 1 hour to 3 hours
Supported NWP data formats	Proprietary formats: NAME fieldsfile (legacy), PP Standard formats: GRIB, NetCDF
NAME topography	Uses orography from driving NWP meteorological model(s) Flat for single-site meteorology
Advanced flow modelling options	Small-scale terrain module (LINCOM); isolated building module
Output quantities	
Model outputs	Two-dimensional fields; vertical cross-sections; location-specific time series; particle trajectory information; model diagnostics
Output quantities	Standard dispersion quantities: air concentration, deposition, cloud gamma dose Meteorological and flow variables Gridded Eulerian fields Other quantities: particle numbers, travel times, plume depth, etc.
Statistical processing	Time averaging/integrating; ensemble averaging; percentiles and probabilities
Data formats	Plain text ASCII file format with user-configured flexible layout Offline conversion to GRIB and NetCDF
Graphical products	Offline IDL and Python (IRIS) utilities to create ps, png, gif images Offline GIS products (ArcView)
Technical characteristics	
Programming language	Fortran 95
Parallelisation	OpenMP for use on multi-core shared-memory systems
Supported operating systems	Linux, Windows
Non-supported systems	NAME has also been built on SUN Solaris and Mac systems
External library packages	ECMWF GRIB API is required when reading GRIB format NWP data (Linux only) NetCDF API is required when reading NetCDF format NWP data

Table 1.1: Overview of NAME characteristics and features (cont.)

1.2 Core Model Infrastructure

This section considers some of the basic model components and infrastructure underpinning the functionality of the NAME model. This includes taking a look at how NAME handles dates and times, the horizontal and vertical coordinate systems supported by the model, and how grids and domains are represented in NAME.

1.2.1 Physical Constants and Units

Table 1.2 lists basic physical constants and other physical parameters defined in NAME, along with the values assumed for these constants. As a general rule, all units in NAME are S.I. Units (unless stated otherwise). One exception here is in the use of feet as a vertical coordinate system when producing aviation products based on the ICAO standard atmosphere.

Physics parameters		
g	$= 9.80665 \text{ m s}^{-1}$	Acceleration due to gravity
N_A	$= 6.02214 \times 10^{23}$	Avogadro's number (molecules per mole)
R	$= 287.05 \text{ J kg}^{-1} \text{ K}^{-1}$	Gas constant for dry air
c_p	$= 1004.6 \text{ J kg}^{-1} \text{ K}^{-1}$	Specific heat capacity for dry air
r_E	$= 6371229.0 \text{ m}$	Mean earth radius
κ	$= 0.40$	Von Karman's constant
m_{air}	$= 28.966 \text{ g mol}^{-1}$	Molecular mass of dry air
m_{water}	$= 18.016 \text{ g mol}^{-1}$	Molecular mass of water
$T_K _{0^\circ C}$	$= 273.15 \text{ K}$	Thermodynamic temperature at zero degrees Celsius
P_{Ref}	$= 100000.0 \text{ Pa}$	Reference pressure for potential temperature
1 Foot	$= 0.3048 \text{ m}$	One imperial foot

Parameters defining the ICAO standard atmosphere		
$T _{0km}$	$= 288.15 \text{ K}$	Temperature at 0 km above mean sea level
$T _{11km}$	$= 216.65 \text{ K}$	Temperature at 11 km above mean sea level
$T _{20km}$	$= 216.65 \text{ K}$	Temperature at 20 km above mean sea level
γ_{0-11km}	$= 0.0065 \text{ K m}^{-1}$	Lapse rate from 0 to 11 km above mean sea level
γ_{20km+}	$= -0.001 \text{ K m}^{-1}$	Lapse rate at more than 20 km above mean sea level
$p _{0km}$	$= 101325.0 \text{ Pa}$	Pressure at mean sea level

Table 1.2: Physical constants defined in NAME

1.2.2 Times

There are two types of time information supported by NAME: *absolute times* and *relative times*. Absolute times are used to represent actual calendar dates and times, such as 01/01/2012 00:00 (i.e. midnight on Jan 1, 2012), whereas relative times are used to specify time offsets or time intervals, such as 12hr 00min (i.e. a 12-hour time period).

Absolute times in NAME take the general form

`dd/mm/yyyy HH:MM[:SS[.sss]] [UTC [±hh:mm]]`

where terms in [] are optional components. Here `dd/mm/yyyy` is a date in the standard Gregorian calendar, `HH:MM:SS` is a time on the 24-hour clock (where the seconds `SS`, with possibly fractions of a second `sss`, are optional) and `UTC ±hh:mm` gives an optional time zone in which the time is expressed. In the absence of a supplied time zone, the default time zone is set to be UTC (in which case, the string 'UTC' itself can be absent). The value `±hh:mm` following the 'UTC' is specified as hours and minutes *ahead of UTC* (so that time zones to the east of the Greenwich meridian will tend to have a positive value and time zones to the west will tend to have a negative value). Note that the time zone minutes are included as well as the hours because, although most time zones are a whole number of hours ahead or behind of UTC, there are some countries which use a half-hour time zone. Note also that any leading zeros in the components of the date `dd/mm/yyyy` or time `HH:MM:SS` can be omitted. As an example of the format, the time `2/8/2012 15:30 UTC+01:00` represents the time 3:30 pm on August 2, 2012 expressed in the time zone 'UTC + 1 hour'.

Relative times are used to define a time relative to some reference time (usually the start of the model simulation) or for representing a time interval in NAME (e.g. the duration of a source release). Two forms are supported for representing relative times: a descriptive text format

`[-][Nday][HHhr][MMmin][SS[.sss]sec]`

and a non-descriptive concise format

`[-][Nd]HH:MM[:SS[.sss]]`

where terms in [] are again intended to be optional components. In these formats, `N` refers to the number of days in the time interval, and `HH`, `MM` and `SS` to the hours, minutes and seconds, respectively. As with absolute times, fractions of a second `.sss` are also optionally supported. A relative time can be negative, in which case a '-' prefix is attached to it, however time intervals are usually defined to be positive. For the descriptive format, at least one of the components corresponding to 'day', 'hr', 'min' or 'sec' must be present. It is permitted to specify the value of a component to lie outside its natural range (e.g. seconds in the range 0 to 59), provided that that component is the most significant non-zero element of the time. For example, a time interval of 300 seconds could be equivalently represented as `00:00:300` or as `00:05:00`, whereas a time period of 72 hours could be written as `3d 00:00` or as `72:00`.

There are also two special time values of 'infinity' and '-infinity' to represent times in the infinite future and infinite past, respectively. These values are useful for defining infinite or semi-infinite time periods, e.g. in specifying an ongoing release from a continuous source or an unbounded temporal domain.

For a more detailed discussion on the handling of dates and times in NAME, including the full range of possible options and the rules associated with these entities, see the [▶ INPUT DOCUMENTATION ◀](#).

1.2.3 Arrays

A one-dimensional array (or ordered collection of values) can be defined in NAME using an [Array](#) block in the input file. Each array block has an associated name, and multiple arrays may be specified in an input file by assigning each one a unique name. Intrinsically, an array in NAME is a collection of character-valued entities, but these array entries often specify numerical values. Arrays are typically used when defining vertical coordinate systems and vertical grids (see [VERTICAL COORDINATE SYSTEMS](#) in 1.2.5.2, and [VERTICAL GRIDS](#) in 1.2.6.3), but can also be useful for other purposes. For example, an array of folder names can be listed to specify the multiple met folders to be used when considering an ensemble of meteorological realisations (see [USING ENSEMBLE NWP METEOROLOGICAL DATA](#) in 1.6.2.2).

1.2.4 Particle size distributions

A particle size distribution can be described using a [Particle Size Distribution](#) block in the input file. As with arrays, each particle size distribution has an associated name, and multiple distributions can be defined in an input file by assigning them unique names. Within a [Particle Size Distribution](#) block, there are five recognised column keywords: [Diameter Range Boundary](#), [Cumulative Fraction](#), [Representative Diameter](#), [Particle Density](#) and [Particle Shape](#).

The [Diameter Range Boundary](#) gives values defining the maximum diameter (in μm) of the particles in each of the size-range bins, and [Cumulative Fraction](#) specifies values for the fraction (defined as a value between 0 and 1) of particles having diameters less than or equal to each of those given diameter boundaries. These first two keywords are used when defining a size distribution for releasing particles (via a [Sources](#) input block).

The [Representative Diameter](#) values are relevant when representing sedimenting material on advected Eulerian fields (via a [Species Uses](#) input block) and allows a representative diameter to be specified for the particles in each size range. If a representative diameter is not specified by the user, the geometric mean of the diameter boundaries will be used. The [Representative Diameter](#) keyword is ignored for particle releases.

The [Particle Density](#) and [Particle Shape](#) keywords are used to define the density and shape of the particles in each particle size range bin. To invoke [Particle Shape](#) the user must also specify the [Sedimentation Scheme](#) they wish to use in the [Sources](#) input block. The prescribed values of [Particle Density](#) and [Particle Shape](#) are used to calculate the sedimentation velocity of the particles. It should be noted that when using the Eulerian scheme particles are assumed to be spherical.

When defining a particle size distribution, the first row must specify a lower-bound diameter for which the fraction of smaller particles is zero (i.e., the specified diameter in the first row is a minimum diameter). Similarly the last row must specify an upper-bound diameter for which all the particles are below the specified diameter (i.e., the specified diameter in the last row is a maximum diameter). Also note that the i -th density and representative diameter correspond to the size range

between the i -th and $(i + 1)$ -th values of the diameter range boundaries.

An example of a particle size distribution for releasing particles is shown below.

Particle size distribution:	Volcano A
Diameter Range Boundary,	Cumulative Fraction
0.1,	0.0
0.3,	0.001
1.0,	0.006
3.0,	0.056
10.0,	0.256
30.0,	0.956
100.0,	1.0

Table 1.3: Example of a particle size distribution in NAME

In this example defining a particle size distribution called 'Volcano A', there are 6 particle size bins within the distribution. The smallest size-range bin has particles ranging from 0.1 to 0.3 microns in diameter, with particles released from this size range at a relative frequency of 0.001 (i.e. one particle per thousand will be selected from this bin). The next bin has particles ranging from 0.3 to 1 micron in size and these particles will be released with a relative frequency of 0.005 (5 particles per thousand), so that the cumulative fraction up to 1 micron is 0.006, and so on. The largest size-range bin will release particles between 30 and 100 microns, and all released particles will have a diameter that is less than or equal to the upper bound of 100 microns.



For earlier versions of NAME prior to v6.4, the column header **Diameter** was used to specify the maximum diameter of particles in each size-range bin. This has been replaced by the header keyword **Diameter Range Boundary**, which will require users to update older input files if they use particle size distributions.

Currently, a maximum of 10 diameter values can be specified within each particle size distribution, and a maximum of 5 particle size distributions can be defined within a set of input files for a single NAME run. For sea-salt sources, only three diameter values (defining two particle size bins) are allowed although the specified diameter values are ignored. Instead the diameter values 0.1 μm , 3.2 μm and 5.0 μm are hard-wired for sea-salt.

Particle size distributions are used in NAME for characterising a size distribution of particles released by a source term (where the release involves a range of different particle sizes), for requesting size-segregated output when modelling particulate material (where the size distribution is used to group the material into particle size bins), and also for representing sedimenting material on advected Eulerian fields (where the size distribution is used to group material into particle size bins which are held on separate fields).

When used to characterise the size distribution of a source term, each model particle released by that source is allocated an individual diameter determined from the particle size distribution. In

general, the diameter is determined such that the logarithm of the diameter is uniformly distributed within each size range with the appropriate proportion of particles (and mass) released within each size bin. Special rules apply for dust and sea-salt sources. For a dust source, a unique particle diameter is associated with each size bin, using the mid-point (on a logarithmic scale) within its specified diameter range. The diameter values specified here should correspond with those of the input soil fraction size ranges. For a sea-salt source, there are two fixed particle size ranges, and a random diameter is determined from a polynomial distribution within each of these diameter ranges. For both dust and sea-salt, the specified cumulative fraction is not used. Instead the mass released in each size range is determined using the input meteorological and land surface data.

When used to characterise sedimenting material on fields, a particle size distribution is applied to group the material into particle size bins, with each bin stored on a separate field. This allows the different sedimentation characteristics of each size range to be represented, using the particle density and representative diameter to determine the appropriate sedimentation velocity for each size range. The [Cumulative Fraction](#) information plays no role in controlling the sedimentation of fields and is ignored when a particle size distribution is used in this way. Note also that the particle size distribution used to control the grouping of material into bins for fields can be, but doesn't have to be, the same distribution used to control the release of particles. The same applies when using a size distribution to characterise the particle size bins for size-segregated output quantities.

1.2.5 Coordinate Systems

There is support in NAME for a variety of generic coordinate reference systems, as well as specific instances of these systems being available to the user in a pre-defined selection of commonly used coordinate systems. For example, NAME can recognise generic latitude-longitude coordinates, possibly using a rotated latitude-longitude reference frame (as is often used by a limited area NWP model) or a latitude-longitude system with an offset origin or a non-standard unit scaling. To interpret such generic coordinates, the user would need to specify the appropriate parameters of the latitude-longitude reference system. However locations are usually expressed in the 'standard' latitude-longitude coordinate system (that is, coordinates referenced as degrees East or West of the Greenwich Meridian and degrees North or South of the Equator) and so a 'Lat-Long' system is also available as one of the pre-defined coordinate systems in NAME.

Note that any coordinate conversions that are necessary during a NAME simulation (for instance, because a source location is given in a different coordinate system to that used by the meteorological input fields) will be handled automatically by the model as and when required. Therefore the user should not be too concerned if several different coordinate systems are involved for various aspects of their model set up – NAME will quite freely handle this complexity. However it should be borne in mind that there may be accuracy issues associated with converting between different coordinate systems, especially where approximate conversions are being employed. Further comments

are noted where accuracy of coordinate conversions could be a potentially significant issue.

Consideration of coordinate systems is separated into [HORIZONTAL COORDINATE SYSTEMS](#) in 1.2.5.1 and [VERTICAL COORDINATE SYSTEMS](#) in 1.2.5.2. See [»» COORDINATE SYSTEMS DOCUMENTATION ««](#) for a comprehensive description of coordinate systems in NAME and of the formulae used for converting coordinates between different systems.

1.2.5.1 Horizontal coordinate systems

NAME supports **five generic coordinate systems** for representing horizontal position on the surface of the Earth, see Table 1.4, together with the pre-defined coordinate systems listed in Table 1.5.

NAME adopts a spherical Earth as its base geoid, rather than any more precise ellipsoid description. This is partly for reasons of computational efficiency and simplification, but also ensures consistency with the approach followed in many NWP models, including the Met Office's Unified Model, of a spherical globe. However, as a consequence of the spherical Earth assumption and the approximated conversions between different coordinate systems, non-trivial positional errors can potentially be introduced into locations. For instance, the 'true' UK National Grid coordinate reference system is based on the Airy 1830 ellipsoid and the NAME approximation to that system might typically produce positional errors of the order of several hundred metres when converting to or from a latitude-longitude location reference. Care should therefore be taken to minimise the impact of such issues (e.g. by not requesting model outputs in a UK National Grid reference frame when the source location has been specified as a latitude-longitude position).

Type of coordinate system	Parameters required to define system
Latitude-longitude	Pole Location, Rotation Angle, Origin Offset, Units
Cartesian in Polar Stereographic projection	Tangent Point, Rotation Angle, Origin Offset, Units
Polar in Polar Stereographic projection	Tangent Point, Rotation Angle, Origin Offset, Theta Offset, Units
Cartesian in Transverse Mercator projection	True Origin, Scale Factor, Origin Offset, Units
Polar in Transverse Mercator projection	True Origin, Scale Factor, Origin Offset, Theta Offset, Units

Table 1.4: Generic horizontal coordinate systems supported by NAME

Name of coordinate system	Coordinate type	Description of coordinate system
'Lat-Long'	lat-long	standard latitude-longitude system (units in degrees)
'Lat-Long (Radians)'	lat-long	standard latitude-longitude system (units in radians)
'EMEP 50km Grid'	PS Cartesian	EMEP coordinate system for the EMEP 50km grid
'EMEP 150km Grid'	PS Cartesian	EMEP coordinate system for the EMEP 150km grid
'UK National Grid (m)'	TM Cartesian	UK National Grid system (with units in metres)
'UK National Grid (100m)'	TM Cartesian	UK National Grid system (with 100 metre units)

Table 1.5: Pre-defined horizontal coordinate systems supported by NAME

1.2.5.2 Vertical coordinate systems

In order to resolve vertical position in the atmosphere, NAME supports **six generic coordinate systems** for representing altitude, see Table 1.6, together with the pre-defined coordinate systems listed in Table 1.7.

Vertical position can be expressed directly as a height (either above the local ground or above mean sea level) or as a pressure (since pressure decreases with height in the atmosphere). The pressure may also be expressed indirectly as an equivalent height above sea level based on the ICAO standard atmosphere (this representation is typically used for aviation products, e.g. volcanic ash advisories issued following an eruption of ash into the atmosphere). Height scaling units can be chosen arbitrarily, although S.I. units are typically used (i.e. metres for the height-based coordinates, and pascals for expressing pressure). NAME also supports two hybrid coordinate systems which follow the terrain near to the surface and then gradually ‘flatten out’ to constant-height or constant-pressure surfaces at higher altitudes in the atmosphere. These terrain-following coordinates are known as eta coordinates, and eta coordinate systems are commonly deployed as the native vertical coordinate in NWP models.

Type of coordinate system	Parameters required to define system
Height above ground level	Scaling unit
Height above sea level	Scaling unit
Pressure	Scaling unit
Pressure converted to height above sea level using the ICAO standard atmosphere	Scaling unit
Pressure-based eta coordinate	η -definition (two of η [:], A [:], B [:])
Height-based eta coordinate	Model Top Height, Interface Height

Table 1.6: Generic vertical coordinate systems supported by NAME

Name of coordinate system	Coordinate type	Description of coordinate system
‘m agl’	Height above ground	height above ground level in metres
‘m asl’	Height above sea	height above sea level in metres
‘Pa’	Pressure	pressure in pascals
‘FL’	Pressure as height	flight level (= pressure converted to height above sea using ICAO standard atmosphere and expressed in units of hundreds of feet)

Table 1.7: Pre-defined vertical coordinate systems supported by NAME

1.2.6 Locations, Grids and Domains

This section discusses the specification of geographical locations in NAME (e.g. for representing the position of a source term or the locations for a list of measurement receptor sites), and how model grids and domains are defined and used within NAME. Horizontal, vertical and temporal grids follow

a similar conceptual structure to each other and have broadly consistent syntactic rules. In addition, all locations, grids and domains have a name associated with them when they are defined, so that they can be referenced as needed from other blocks in the input files.

1.2.6.1 Specifying Geographical Locations

There are a number of ways that positional information can be supplied to NAME. For instance, when specifying a source term the user might frequently give three-dimensional coordinates (x, y, z) of its centroid, together with the horizontal and vertical coordinate systems in which those values are expressed. However, to further assist the user in working with location coordinates, NAME functionality includes the ability to define named geographical locations.

One or more named locations may be stored together using a **Locations** block in the set of main input files. Such a block might be used, for example, to store the coordinates of potential source locations (e.g. a list of nuclear power plants) or a collection of receptors (e.g. an air quality measurement network). Each location in a collection is assigned a name and is defined by specifying its two-dimensional position (x, y) in a given horizontal coordinate system. In principle, a collection of locations might include geographical locations defined in a variety of different coordinate systems, but in practice they are usually all specified in the same system (and indeed some uses of locations within NAME require this to be the case).

A set of locations itself has a name, so that it can be referenced from elsewhere in the input files, and it is quite possible for there to be multiple sets of locations defined in the input files for various purposes (provided that each set has been given a unique name). Locations can be referenced individually (to identify the position of a source or the centre of a domain, say) or collectively as a set (e.g. when identifying a list of receptors).

1.2.6.2 Horizontal grids

Horizontal grids are specified in NAME using a **Horizontal Grids** block in the set of main input files. They can be defined in a number of different ways depending on requirements and are used for a variety of modelling purposes. For instance, horizontal grids can be *unstructured* (i.e. a set of points with no underlying geometrical structure) or *structured* (following a rectangular 2-d lattice pattern). Structured grids can themselves be divided into those with a regular gridpoint spacing (i.e. a constant dX and dY) and those with irregularly-spaced lattices. Figure 1.1 depicts these various types of horizontal grid in schematic form. Note that although partially supported by NAME, not all NAME functionality is available for irregularly-spaced structured grids and they are not often used.

An unstructured grid (Figure 1.1a) is created by ‘attaching’ a grid box at the positions defined by a set of locations (see 1.2.6.1). The size of every such grid box is specified by supplying a pair of values (dX, dY) together with the horizontal coordinate system in which these values are expressed. An unstructured grid might be used, for example, to request model outputs such as time

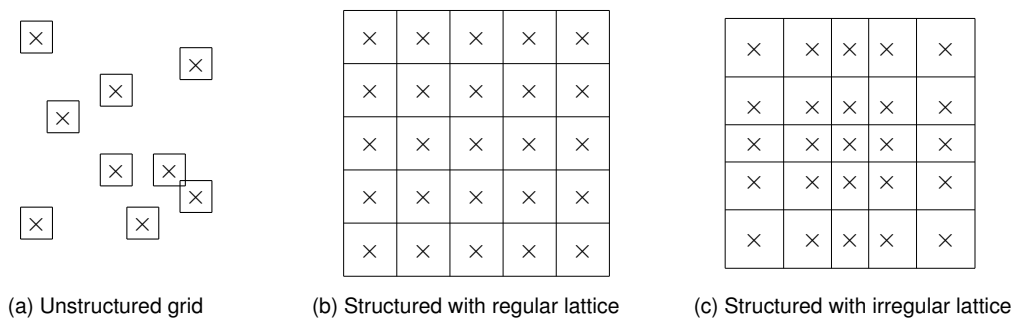


Figure 1.1: The types of horizontal grid that can be defined within NAME

series at those locations.

For structured grids, the grid boxes are defined implicitly using the edges equi-distant between grid points: with a regular grid (Figure 1.1b), all grid boxes have an equal area, whereas irregular grids (Figure 1.1c) possess grid boxes that vary in size. The variation in grid size of irregular grids is one reason why they are not fully supported in NAME. Structured grids can be specified in a multitude of different ways depending on the user's preference. But the basic information that is required are one or more geographical positions to geo-reference (locate) the grid in space and an indication of the number of grid points and/or the grid point spacing in each direction. See the [INPUT DOCUMENTATION](#) for further details about specifying grids.

The interpretation of gridded values depends on the quantity that is being considered. Gridded data might represent 'spot values' at those grid points for some parameters, or they may represent grid-box average values evaluated over the volume of the grid box (after a suitable vertical averaging has also been specified, see 1.2.6.3).

Horizontal grids have various uses within NAME, but the most common uses are in the specification of input NWP meteorological fields and in requests for model outputs from NAME. For the former, gridded NWP met data are always read on a structured grid having a regular gridpoint spacing, whereas model outputs may often be requested on both regular grids (for 'fields') and at an unstructured set of locations (for 'time-series').

1.2.6.3 Vertical grids

Vertical grids are specified in NAME using a [Vertical Grids](#) block in the set of main input files. Vertical grids are used to specify information about height levels or a vertical discretisation in the height coordinate. As with horizontal grids, vertical grids can be defined by the user in various ways, with four examples shown in Figure 1.2 of types of vertical grid that can be used in NAME.

A uniform vertical grid (Figure 1.2a) is defined by specifying the underlying vertical coordinate system to use (i.e. the ruler by which to measure heights), together with the height Z_0 of the first (lowest) grid point, the spacing dZ between each pair of adjacent grid points and the total number nZ of points in the vertical grid. Vertical averaging, if relevant, is centred on each grid point and

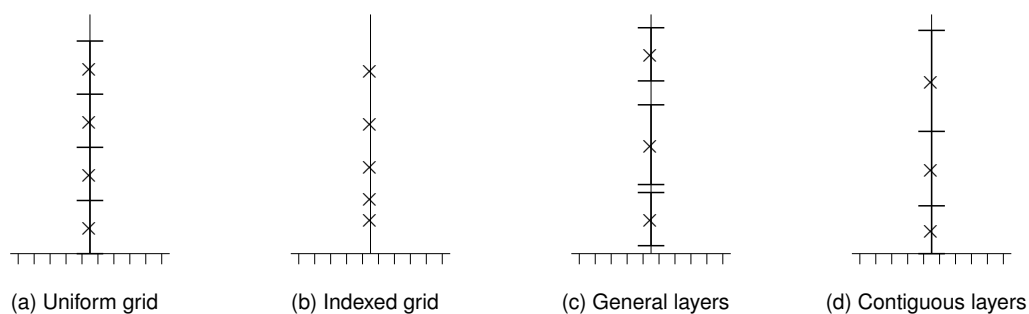


Figure 1.2: Examples of vertical grids that can be defined within NAME

also applies the same dZ value. Although a vertical grid will often contain multiple grid points (i.e. $nZ > 1$), this does not necessarily need to be the case and vertical grids with a single point ($nZ = 1$) are just as valid in NAME. Such single-point grids can be used to request output, say, for a single layer within the atmosphere. For example, an average evaluated over the height range 0 – 100 metres can be requested in NAME using the pair of values $Z_0 = 50.0$ (the mid-point of the layer) and $dZ = 100.0$ (the depth of the layer) in a ‘metres above ground level’ coordinate system.

Non-uniform grids can also be constructed in NAME and may be defined with or without vertical averaging information (i.e. as a set of discrete grid points (vertical levels) or as vertical layers). A grid comprising of discrete vertical levels can be specified using an array, $ZArray$, to give the values of the vertical grid points referenced with respect to a specific vertical coordinate system, as in Figure 1.2b, with the possible option of declaring a second indexing array of integer values to label those vertical levels (such a grid is then referred to as an indexed grid). An indexed grid of vertical levels is often defined when utilising NWP meteorological fields in NAME. Alternatively, additional averaging information can be supplied to define a set of vertical layers as in Figure 1.2c, where two arrays, Z and AvZ , are used to prescribe the mid point and depth, respectively, of each vertical layer within the grid. It is not necessary for the layers defined by this approach to be mutually adjacent and it is also possible for such layers to overlap if desired. Finally, a grid comprising of contiguous vertical layers (which may or may not have uniform depths) can also be specified by giving the ‘interface’ levels between layers, as in Figure 1.2d.

As with horizontal grids, interpretation of gridded values depends on the quantity being considered and data might represent ‘spot values’ at grid points or vertically-averaged values evaluated over the depth of the layer.

1.2.6.4 Temporal grids

Grids can be defined in the time dimension in a conceptually similar manner to the definition of spatial grids. Time grids are typically used in NAME to define the times at which output quantities are required. Temporal grids are specified using a **Temporal Grids** block in the set of main input files, as illustrated by Figure 1.3.

Temporal Grids:
Name, nT , dT , T_0
TGrid1, 6, 01:00, 01:00 01/01/2000

Figure 1.3: Example of an input block defining a time grid in NAME

A temporal grid is a specification of an ordered sequence of discrete times (see [TIMES](#) in 1.2.2), which may be regularly or irregularly spaced in time (although regularly spaced time grids are nearly always used). A regular time grid is specified by giving the first time T_0 in the sequence of times, the number of points nT in the time grid, and the time interval dT between each pair of adjacent times. So, for example, an hourly grid as specified in the above example with its first time at 1 am on 1 January 2000 and its last time at 6 am on that same day would be depicted as shown in Figure 1.4.

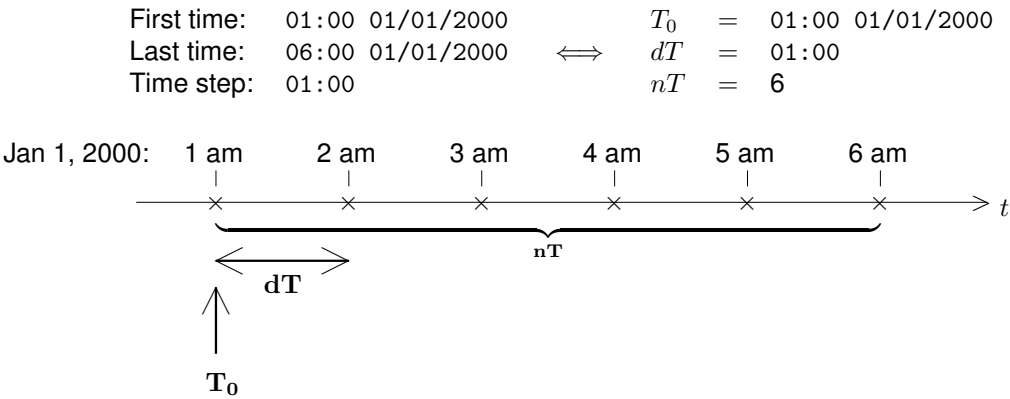



Figure 1.4: Example of a regular time grid (as defined in Figure 1.3)

 An irregular time grid may be specified by providing an array of times (see [ARRAYS](#) in 1.2.3). However this option is not fully supported in NAME and it is advisable to use regular time grids only (which, in practice, are sufficient for the demands of virtually all NAME users).

1.2.6.5 Domains

A NAME domain is a specification of a spatial and temporal region of validity, and is defined using a [Domains](#) block in the set of main input files. Domains are primarily used for two purposes: to constrain the region in which particles are followed within NAME (the computational domain), and to specify a region of validity of flow information (a flow domain).

Various aspects of a domain can be constrained:

- horizontal extent
- vertical extent
- temporal extent
- travel time of particles

For instance, the horizontal extent may be unbounded (i.e. infinite) or bounded to a geographical area. Information on geographical bounds can be supplied in various ways that generally follow a similar syntax to that used for specifying horizontal grids. For example, it is possible to specify minimum and maximum coordinate bounds in a given horizontal coordinate system, or to specify a central location with a range of values about this point. See the [» INPUT DOCUMENTATION «](#) for further details about specifying the horizontal extent of domains.

Similarly, the vertical extent of a domain may be unbounded or may be bounded by specifying a maximum height in the relevant vertical coordinate system. The temporal validity of a domain can also be constrained, if required, by specifying an appropriate start time and end time (or just one of these times, together with a period of duration). Either of the start time or the end time may be in the infinite past or the infinite future, respectively, to give us infinite or semi-infinite time periods. This requires use of the two special time values of ‘-infinity’ (for the infinite past) and ‘infinity’ (for the infinite future). Constraining the end time of the computational domain can be an effective method for specifying the time at which the model run finishes (this will be the earlier of the domain end time and the last time in all temporal grids used by the output requests). Finally, it is also possible to constrain the lifetime of particles within a domain, such that particles exceeding a certain age will expire and be removed from the simulation.

The computational domain in a NAME run specifies the region in space and the time period for which particles will be followed by the model. If a particle leaves this spatial region, either by crossing one of the lateral domain boundaries (a side wall) or by exceeding the maximum permitted height (the vertical lid), then this particle will be ‘killed off’ and removed from the model calculations. Particles are also removed if they reach the end time applicable to that domain or once their age exceeds a specified time limit. The latter method, in particular, is an effective means of removing from the model simulation older particles that would no longer have a significant contribution to the results. The ability to remove particles from the simulation, either by constraining the geographical extent of consideration or by limiting their age, is especially important for NAME runs covering longer time periods. Otherwise the total number of particles in the model simulation might steadily increase as more and more particles are released from the sources. This would gradually increase the memory requirements for the simulation but would also lead to a progressively slower run time. Therefore, for a multi-year chemistry run, for instance, one would constrain the geographical domain (e.g. to the European area), while for a volcanic ash run one might consider limiting the age of ash particles to one week, say.

The other common use of domains is in specifying the validity of flow information, e.g., from that provided by an NWP model. A flow domain is associated with each set of input meteorological data supplied to NAME and specifies its spatial coverage and time period of validity. For example, a forecast from a limited area NWP model would have a bounded horizontal extent and may only provide meteorological information out as far as 48 hours into the future. Whereas a global model would provide meteorological data globally and perhaps a forecast out to six or ten days.

1.2.7 Advanced modelling options and controls


NAME can be configured to run in various modes of operation, which are selected using some high-level control options. These model switches and options are described in the current section and enable a broad range of functionality to be accessed as and when required for a particular modelling application.

1.2.7.1 Restartable NAME runs

Normally NAME runs from a 'cold start'. That is, no pre-existing information from any previous model run is used in starting the current simulation. Instead the model atmosphere is initially empty of particles and any internal fields are initialised to zero. As the simulation progresses, NAME then proceeds to populate its atmosphere with particles released by the source terms, and to calculate the required output quantities and write these to the output files. This approach is suitable for most NAME applications. However there are situations where it is desirable to have the ability to 'save' the state of a NAME run in order for it to be used as the starting state for a new simulation.

One set of circumstances in which this capability to 'restart' a NAME simulation from a saved model state can be useful is in protecting against computer problems or power failures during a very long simulation. For example, a multi-year air quality type of run modelling chemistry over a wide region can require several months of computer processing time. It would be very frustrating if the NAME run failed during the final few days of such a run, say because of a power outage or because of a network problem in accessing NWP met data on a remote disk. This sort of issue can be guarded against by writing a restart file at a regular interval during the NAME run (e.g. at the end of each simulated month). Then, if the worst happens and the NAME run were to fail, the user would only need to restart from the last saved state.

Another situation in which the writing of a restart file can be useful is where the model needs to be regularly started from an initial 'spun-up' state. For instance, a daily air quality forecasting system might utilise this capability. Here the emissions that have occurred over the days prior to the nominal model start time (and the influence that atmospheric processes such as chemistry and deposition have had on those emissions) are already captured in the initial model state. Commencing from a restart file in this way therefore avoids the need to have a 'spin-up' period in the NAME simulation every day. Instead, a short simulation using analysed NWP meteorological fields can be performed each day and that model state saved for initialising the next run on the following day.

 Users should be aware that there are overheads when writing restart files, both in terms of data storage volume (WARNING – the files can be very large!) and in terms of the computational overhead slowing down the NAME simulation (although the impact on model run time is likely to be fairly limited, at least in fractional terms, provided that restart files are not being requested too frequently). However, unless there is a specific need for restart files, the general advice is that they should not normally be requested in a NAME run.

Restart functionality is invoked using a combination of command-line options and other information specified via the input files. Restart files are written during a NAME run according to the instructions supplied in the [Restart File Options](#) block of the input files. Typically these variables are left blank, indicating that no restart files are to be produced. Alternatively, the frequency at which restart files are written can be specified either as a number of cases (see [RUNNING MULTIPLE CASES](#) in 1.2.7.5 below) or as a time interval within each separate case. The individual restart files are labelled by the point in the NAME run (i.e. case and time) at which they are created. Other options for handling restart files can also be included here. For instance NAME can automatically delete old restart files once a newer one has been written. It is also possible to request that a restart file is created whenever the NAME run is suspended (see [SUSPENDING A NAME RUN](#) in 1.2.7.6).

By default, restart data are written to the folder containing the main input file. As restart files can be large, this behaviour may not always be appropriate (for instance, one would usually prefer to write restart files to a local disk rather than to a network drive). Therefore NAME allows the user to specify a separate restart folder using the command-line argument `-RestartFolder restart_folder`. The restart data consist of the restart files themselves (where each file contains a description of the model state from which to restart at a specified time), restart catalogues (containing information on the available restart files), and restart catalogue validity files (containing information on the validity of the restart catalogues).

NAME can be instructed to restart from a restart file by specifying the command-line argument `-Restart restart_identifier` with the (optional) argument value indicating the point in the NAME run at which to restart (i.e. the restart file to be used for the restarted run). If the optional value is omitted, the latest restart file will be used (and in the event that there are no restart files available then the run will start from the beginning). In the absence of the `-Restart` option, NAME will start the run from the beginning. However, restart catalogue validity files should not be present without the `-Restart` option and NAME checks to ensure these files do not exist, otherwise the run stops to prevent accidental overwriting of previous output files.

For further information on restarting NAME runs, see the [INPUT DOCUMENTATION](#).

1.2.7.2 Repeatable NAME runs

The use of random numbers is an intrinsic feature of any random-walk model, and so NAME has a fundamental requirement for sequences of random numbers. NAME produces these random sequences using its own random number generator, rather than using the intrinsic Fortran random number generator. In fact, use of the intrinsic random number generator is available as a deprecated option in NAME, although we would generally advise against its use. See separate technical documentation for further details of the random number generator implemented in NAME.

From a user perspective, control of the random number generator is achieved through the variable [Random Seed](#) in the [Main Options](#) block of the main input file. This variable can have one of four possible values: *Fixed*, *Fixed (Parallel)*, *Random* or *Random (Parallel)*. Here the *Fixed* and

Random options are deprecated in favour of the *Fixed (Parallel)* and *Random (Parallel)* options.

By selecting *Fixed (Parallel)*, NAME will use a repeatable, identical sequence of random numbers every time a run is performed. Therefore, in the absence of any change in the model inputs, the outputs from two separate NAME runs should be identical.¹ That is, the model run is *repeatable*. This functionality can be useful when tracking down any problem encountered during a model run (as the scenario can be recreated later in a subsequent run). The alternative option of *Random (Parallel)* uses fully-randomised random numbers, which start from a different seed in every NAME run. This yields different (non-repeatable) random sequences and therefore separate NAME runs with the same inputs will not produce identical outputs. In other words, the model run is *non-repeatable*.

There is also a variable *Same Results With/Without Update On Demand?* in **Main Options** that regulates how met and flow module instances are updated, and which should be set to *Yes* to indicate that the same results are required whether ‘update-on-demand’ is used or not. This ensures the same met update procedure is followed when using update-on-demand, and that particles use consistent synchronisation steps.

1.2.7.3 Running NAME backwards in time

For most applications NAME is run forwards in time to represent the transport and evolution of material released from a source. However NAME can also be run backwards in time to model the history of material arriving at a receptor. When run in backwards mode (also referred to as adjoint mode), the simulation is often termed a NAME back-run. NAME back-runs are designed to represent air history and because of this the output products are often called air history maps. They describe where the air at a receptor at any given time has been over the preceding hours and days, during which time that air parcel might have become contaminated with pollutants. The air history can therefore provide useful insight into identifying the origins of an air parcel and possible contributory sources for polluted air (e.g. during a pollution episode).

When run in adjoint mode, the backwards spread of an air parcel is still represented in a diffusive sense (as in a forwards run) with the diffusion representing mixing of background air into the air parcel by turbulent processes. This is somewhat different to ‘just running the dispersion model backwards’ where spread of the plume decreases with time (as one might envisage, say, by running a film backwards). Spread is always expected to increase with larger travel time (whether this is in a forwards or a backwards sense). Also note that NAME has limited functionality when run backwards and some processes cannot be represented in a back-run (this is, in part, due to the irreversibility of many of the processes represented by NAME). Back-runs are generally restricted to consideration of a tracer in a basic particle run. For example, processes such as radiological decay and chemistry are ignored. Output requests are also typically restricted to the time-integrated air concentration for a back run, because this quantity provides the integrated ‘footprint’ of where the air has been over

¹Strictly speaking, small differences may still occur between the output fields produced by two *parallelised* NAME runs because of the non-associative nature of finite-precision arithmetic (due to the influence of numerical precision).

the duration of the model run.

NAME is configured to run backwards in time by setting the keyword [Backwards?](#) to [Yes](#) in the [Main Options](#) block of the main input file. Apart from setting this Backwards switch, the user also needs to be careful when defining objects that involve times, especially in specifying the start time and end time of the computational domain and the start and stop times of any source terms. These time periods are always set relative to the direction of time in the model run, and so need to be defined accordingly when time runs backwards. In a chronological sense, this implies a start time that is later than an end time for domains, and similarly a start time for a source that is later than its stop time. This is one of the potentially confusing aspects with back-runs, and it is advised to expend a little effort in ensuring that times are defined correctly according to requirements. For time grids, the [T0](#) value still refers to the first time point in the grid, which in the case of a backwards run is (chronologically) the latest time in the grid, and the time interval [dT](#) is expressed as a positive time interval (even though time is running backwards).

1.2.7.4 Using ‘fixed meteorology’ in NAME



At the present time, the ‘fixed met’ option in NAME is under development and it is recommended not to use this scheme.

The intention with the ‘fixed met’ option is to enable NAME to treat short-range dispersion problems using a meteorology that is fixed in time (that is, a stationary flow field). Under this scenario, particles and/or puffs can be released at a single time only, and the effect of continuous or finite-duration releases can be treated by integration over measurements times (by using the stationarity of the flow). However the scheme is not yet fully implemented and may not produce correct results.

1.2.7.5 Running multiple ‘cases’

In most applications there is usually only a requirement to consider a single realisation, or ‘case’, of a dispersion simulation. However, it is possible to consider multiple cases within a single run of the NAME model. That is, one can launch a NAME run in which a scenario is repeated under different options or circumstances to generate a sequence of multiple realisations. For instance, NAME can re-run a scenario for different choices of the dispersion modelling options (perhaps to compare a simulation using particles with one based on puffs, or a simulation considering the effect of deposition with one that does not). NAME can also consider multiple realisations of the input meteorology from an NWP model, e.g. the different ensemble members provided by an ensemble prediction system.

Multiple cases are invoked according to the instructions supplied in the [Multiple Case Options](#) block of the main input file. Two variables may be specified via this block: [Dispersion Options En-](#)

[semble Size](#) and [Met Ensemble Size](#). Typically these variables are left blank or are both set to one, indicating that NAME should only consider a single realisation based on a single set of dispersion model options and a single met realisation. Alternatively, one or both of these values may be set to a value greater than one to indicate that multiple cases are to be considered. Usually multiple cases are considered for either the model options or the meteorology separately, although there is no reason why both options cannot be specified together, in which eventuality each met realisation is repeated for each set of dispersion model options (so that if there are m sets of dispersion options and n members of the meteorological ensemble, this will give rise to $m \times n$ realisations of the dispersion). If an ensemble of dispersion options is requested here, the appropriate number of sets of dispersion model options must also be listed in the [Sets of Dispersion Options](#) block of the input file.

For further information on specifying dispersion model options, see [DISPERSION MODEL OPTIONS](#) in 1.7.1. For further details about using ensemble met forecasts, see [USING ENSEMBLE NWP METEOROLOGICAL DATA](#) in 1.6.2.2.

1.2.7.6 Suspending a NAME run

A NAME run can be suspended (i.e. terminated) once the model has passed a specified point in the simulation (e.g. a specified end time). The suspension of a NAME run can be requested when the NAME run is launched, using the command-line options `-RunTo time | case` or `-RunFor time interval | number of cases`, or once NAME is running via the use of a 'run-to' file.

The value supplied with the `-RunTo` argument instructs NAME to suspend the model run at the specified model time or after the specified case has completed. Alternatively, the `-RunFor` argument can be provided, in which case the NAME run terminates after the specified model time interval has elapsed or after the specified number of cases has been completed. Note that the format for times and time intervals used here follows the general formatting rules for times (see [TIMES](#) in 1.2.2), but with spaces being replaced by under-score characters.

NAME can also suspend a model run in a dynamic way by monitoring a 'run-to' file. This approach enables a running simulation to be stopped as required, even if the suspension criterion is not initially known when the run is launched. If this ability to control the model run is required, the name of the 'run-to' file to be monitored must be specified in the [Run-To](#) variable in the [Main Options](#) block of the main input file. If at any time during the run the 'run-to' file is created, then the file contents are read and, if the file is empty or contains anything other than a time, the run is suspended immediately, while, if the file contains a time, the run is suspended once that time is passed.

1.2.7.7 Parallelisation options

NAME can be run in a multi-threaded environment on modern shared-memory multi-processing computer systems. Certain sections of the NAME code have been parallelised using OpenMP directives to instruct the runtime environment to assign computational tasks among multiple threads (or processes). Parallelisation of the code in this way is designed to speed up model run times by utilising additional processor capacity that may be available on the computer.

The present section is concerned with how a user configures a parallel NAME run. See [PARALLELISATION WITH OPENMP](#) in 1.8.4 for technical details concerning the parallelisation of NAME using OpenMP. See also [MD18/1: PARALLELISATION WITH OPENMP](#) for further information about NAME parallelisation in general.

A parallelised version of the NAME executable should be used when parallel functionality is required by the user, see [COMPILING NAME](#) in 1.8.2 for details. It is recommended to set the environment variable `OMP_STACKSIZE` when running with multiple threads under OpenMP, e.g.,

```
export OMP_STACKSIZE='4m' on Linux systems, or
setenv OMP_STACKSIZE='4m' on Windows systems.
```

Here the `OMP_STACKSIZE` variable defines the stack size that is allocated for any additional threads that are created. The default, if not set explicitly, is 4 MB ('4m') and this is usually sufficient when running NAME in parallel. However it may sometimes be necessary to set a larger value than this. The NAME run will fail with a 'memory fault' if the stack size limit is insufficient for the run.

The behaviour of the parallel code at runtime is controlled by the [OpenMP Options](#) block in the main input file. The logical keyword [Use OpenMP?](#) indicates whether OpenMP functionality will be used in the NAME run. When set to [Yes](#), certain computational tasks are distributed amongst multiple threads depending on the settings supplied in the other input variables. If not defined, the default is set as [No](#) and all OpenMP directives are ignored (so that NAME will effectively run as a serial process). Other options are available via the remaining keyword variables in this input block.

Keyword	Description
Threads	Number of threads in all parallelised loops (unless overridden for specific loops by the variables below)
Particle Threads	Number of threads in the particle loop
Particle Update Threads	Number of threads in the particle update loop
Chemistry Threads	Number of threads in the chemistry loop
Output Group Threads	Number of threads in the output group loop
Output Process Threads	Number of threads in the output processing loop
Parallel MetRead	Indicates that NWP met data should be read with a separate I/O thread
Parallel MetProcess	Indicates that NWP met data should be processed with a separate I/O thread

The number of threads specified for any loop should be a positive integer (i.e. greater than or equal to 1) and should normally be limited to the number of cores available on the system. The **Threads** keyword provides a global value to be used for all parallelised loops unless this is overridden for any specific loops by the other variables. The global number of threads defaults to **1** if not defined.

The two logical variables **Parallel MetRead** and **Parallel MetProcess** are used to launch a separate dedicated thread for reading NWP met data files from disk and then processing the NWP fields, respectively, in parallel to the main part of the processing performed by the other threads. This enables NWP met data to be read and processed in memory ahead of demand while the rest of the NAME run is still progressing, and therefore for the meteorological fields to be immediately available when required at the next time step.

If neither of these variables is specified, they both take the default value **No**. When **Parallel MetRead** is set to **Yes** and **Parallel MetProcess** is not specified, then the latter is automatically set to **Yes** as well, i.e. the met data is read and processed by the I/O thread. When **Parallel MetProcess** is set to **Yes** then any value given for **Parallel MetRead** is automatically reset to **Yes**.



Note that it is currently not possible to use a parallel I/O thread together with the met-on-demand functionality.

1.2.7.8 Controlling messages, warnings and errors

On execution of a NAME run, various informational messages will be written to the standard output device (i.e. the display screen when NAME is used interactively, or an output file when NAME is run from a script) as the model run progresses. This information is also written independently to two output text files: a 'Log' file contains a copy of all messages sent to the standard output, whereas an 'Error' file lists those messages which are classified as abnormal. These two text files use the same file prefix as the main input file for that NAME run. (Note that there may also be various 'Restart' files or a 'Timing' file depending on the setup of the particular model run.)

Informational messages are classified according to the type of information they contain:

a message A general message, e.g., to inform on progress through the model run.

a warning A report that NAME has encountered something unusual (possibly an inconsistency in the user setup or a problem in the input data). The problem has the potential to impact on quality of model output, but is not serious enough to be classed as an error.

a non-fatal error A report that NAME has encountered an error from which it is able to recover. It is likely that the model output will be degraded in some manner.

a fatal error A report that NAME has encountered a serious error from which it is unable to recover. Some model output is likely to be missing.

an unexpected fatal error A report that NAME has encountered a serious internal error from which it is unable to recover. Unexpected errors are those which will typically require code or system changes to correct them (i.e. they are not due alone to errors in the user input).

Any unexpected error should be reported and our advice sought on how to resolve the problem.

There is some user control over how certain messages are classified and reported. A limited number of messages can be controlled through a [Message controls](#) block in the input file. The keyword pairs [Name](#) and [Action](#) listed in this block specify the appropriate action to be taken for each of the named error messages. This allows the user to change the default action, e.g. to escalate an [Error](#) to a [Fatal Error](#) (and thereby cause the model run to terminate at that point) or to downgrade a [Fatal Error](#) to an [Error](#) (and thereby allow the model to continue running). It is also possible to set additional keywords [# of Messages Before Failure](#) or [# of Messages Before Suppression](#), which allow a certain number of messages of this type before escalation to a fatal error or before the messages are suppressed, respectively.

User controlled error messages currently exist to deal with the following situations (additional controlled messages could be added if a need arises):

‘No flow’ where there are no valid flow modules with the flow attribute (default behaviour is ‘fatal error’).

‘No flow for particle/puff’ where particles/puffs are being lost due to lack of a suitable flow module (default behaviour is ‘error’).

‘Missing NWP field’ where an NWP field which is expected to be present in the met file is missing (default behaviour is ‘warning’).

‘Inconsistent NWP header’ where the header of an NWP field is inconsistent with the information contained in the met definition for that met file type (default behaviour is ‘error’).

‘Inconsistent validity time’ where an inconsistency is found between the validity times of different met fields (default behaviour is ‘error’).

‘Missing radar met field’ where a field expected to be present in a radar met file is missing (default behaviour is ‘warning’).

‘Inconsistent radar met header’ where the header of a radar met field is inconsistent with the information contained in the met definition for that radar met file type (default behaviour is ‘error’).

‘Inconsistent radar validity time’ where an inconsistency is found in the validity time of a radar met field (default behaviour is ‘error’).

‘Zero source strength’ where all source strengths for a source are zero (default behaviour is ‘warning’).

‘No time zone in single site met file’

1.3 NAME Inputs

This section summarises how input data are supplied to the NAME model and the various options that are available to the user to structure their input in the most effective manner.

1.3.1 Filename conventions for NAME

All filenames are case sensitive, both for input files and output files. So, for example, 'MyRun.txt', 'myrun.txt' and 'MYRUN.TXT' would all be interpreted by NAME as distinct input files. The file suffix '.txt' is usually associated with the main input files to indicate that they are plain text (ASCII) files and should be opened as such by an editor. However there is no specific requirement to use this suffix and, in principle, a user could adopt another suffix or not use any suffix in their filenames.

When specifying folder paths (or directory paths in the language of Linux users), NAME will automatically use the appropriate delimiter for its current operating system. That is, it will intelligently switch between using a '\' on Windows platforms and a '/' on Linux or Unix systems. This means that NAME input files are portable between different operating systems and the user can freely choose which delimiter they wish to use.

1.3.2 Command line arguments

While most NAME inputs are provided through its input files, some options are supplied via the command-line. See the [INPUT DOCUMENTATION](#) for further details.

1.3.3 'Headed' input files for NAME

Most of the non-meteorological input to NAME is specified via a set of input files. There is a considerable degree of flexibility in structuring the information within these input files. For example, all the information can be contained in a single input file (called the main input file) or may be grouped across multiple input files (for example, using separate files to define species characteristics, source terms, met data characteristics, etc.). In the latter case, the main input file needs to reference all the additional input files containing information defining the model run by using an [Input Files](#) block.

The user has the freedom to choose whichever approach is better suited to their application but, generally speaking, a single input file tends to be used for static runs of a relatively simple nature, whereas multiple files are used for more complex runs (e.g. where there are lots of sources) or where certain aspects of the model set up need to be easily edited and it is more convenient for this information to be separated out into a separate file or collection of files.

Within each input file, information is contained in a series of blocks, with each block defining a certain aspect of the model set up. Blocks have a particular structure and follow certain syntactic rules. The order in which blocks appear within input files is not important, although it is good practice to define objects before they are referenced by other objects as this usually makes the input file

easier to understand. Any text that is not contained within the structure of a block is treated as a comment by NAME. This means it is easy to add comments in your input files, just ensure that you add a blank line before and after your comment and that your comment does not start with one of the defined block keywords (one way to guarantee this is to use a comment character such as '!' or '#' as the first character in each comment – this also helps to improve the readability of the file). See the [INPUT DOCUMENTATION](#) for further details on setting up input files.

1.3.4 Other files supplying input data for NAME

While most of the user input is specified via the main set of input files, other sources of input data might be used by NAME for various purposes. The most obvious example here is in the provision of meteorological data to the model. Met data might take the form of hourly records of meteorological variables measured at a weather station, or three-dimensional gridded fields from an NWP model. NAME will read such information as and when required during a model run. Further details on the met data requirements of NAME and how NAME reads met data files are included in [METEOROLOGICAL INPUT FOR NAME](#) in 2.1.

Another situation where NAME reads a different type of input file is when using the NAME chemistry scheme. Here chemistry fields from the Met Office global chemistry model, STOCHEM, are read at the start of each month to set the background chemistry fields for the NAME simulation (see [NAME CHEMISTRY SCHEME](#) in 1.7.10).

1.4 NAME Outputs

For most NAME users, model output will be in the form of data files written to a disk on their computer system or network. Output is requested in this way by ensuring that the character *D* (for Disk) appears in the string of characters supplied for the keyword variable [Output Route](#) in the relevant output block ([Output Requirements - Fields](#) for fields or [Output Requirements - Sets of Particle/Puff Information](#) for trajectory-based information). Two further modes of output are also possible but tend to be used infrequently. By including the character *S* (for Screen), numerical output can be written to the screen. This option works best on Windows systems, where the output is written to a separate window (on Linux systems the output is written to the standard output device and can become mixed up with other messages produced by NAME). The user should also be aware that output data streams can potentially be very large, and should therefore be cautious in restricting screen outputs to quantities with a small data volume. A further option of including the character *G* (for Graphical) can also be supplied for generating simple graphical output (e.g. vertical profiles). However this option is only supported on Windows, and the functionality is rather limited. In principle, more than one of *D*, *S* and *G* may be listed in the string provided for [Output Route](#) in an output request in order to output that information by more than one route.

All files directly output from the NAME model are in plain text (ASCII) format², although there are plans to extend the capability to output in binary formats such as GRIB and NetCDF. It is possible, of course, to convert output text files into these formats (and others) via a post-processing step and this is often done at the Met Office.

Outputs from NAME can be separated into two broad types: field-based outputs and trajectory-based outputs. In the former type, model output is generated on user-specified grids and may involve some form of averaging over all particles within a given grid-box volume (for those quantities computed from model particles), while the latter type outputs information directly along model particle trajectories.

1.4.1 Requesting output of fields from NAME

Output fields are requested using an [Output Requirements - Fields](#) block in the set of main input files. Table 1.8 lists the available quantities that can be requested as model output from NAME. These fields are arranged into five distinct categories depending on the nature of the output and its method of calculation: *Lagrangian quantities* are fields computed using information stored on particles or puffs, *Eulerian quantities* are fields held on Eulerian grids, *Combined quantities* are combined fields obtained by summing the contributions from particles (or puffs) and Eulerian fields, *Flow quantities* refer to meteorological or surface characteristics derived from the relevant flow module

²Strictly speaking, NAME also produces restart files (with the restart data being written to file as unformatted binary). However we would not ordinarily regard this as being NAME output, which is the term usually reserved for model output quantities requested by the user such as air concentration or deposition.

(and that are independent of the dispersion problem itself), and *Other quantities* are miscellaneous output fields and model diagnostics that may be useful to the user.

Supplementary information to further qualify a request may be needed when asking for certain fields, see Table 1.9. For instance, in requesting air concentration, the user needs to specify the species (or list of species) for which air concentration is to be calculated and returned as output. As another example, any request for plume height or vertical plume spread needs to be accompanied by a suitable vertical coordinate system in which the height values are expressed.

Where the functional form of a field exhibits a dependence in a specific dimension, then a suitable grid needs to be provided by the user to span that dimension (i.e. a horizontal grid where there is a dependence on x and y , a vertical grid where there is a dependence on z , a time grid where the field varies with t and a travel-time grid for a dependence on s). A notable exception here are the special rules concerning spatial averaging that are discussed below. So, for example, a request for a basic air concentration field (with no special averaging or integrating applied) would require the specification of a horizontal grid, H-Grid (see 1.2.6.2), a vertical grid, Z-Grid (see 1.2.6.3), and a temporal grid, T-Grid (see 1.2.6.4), to define the points at which the field is to be calculated.

A field request may be assigned a name as a means of identifying that output requirement. The name will also be used to label that field in output files. The explicit naming of field requirements is not compulsory, although it is generally recommended that a user assigns a name to each request. If a name is not specified, the model will automatically assign a name internally to the request.

For field quantities that need species information to be provided (such as air concentration), the user should supply the name of a previously declared species (see [DEFINING SPECIES INFORMATION IN NAME](#) in 1.5.1) or, in the case where multiple species are being requested, a semicolon-separated list of species names. It is possible to restrict the calculation of Lagrangian quantities to consider only the contribution from a particular source or group of sources. This can be done by specifying the name of a source (see [DEFINING SOURCE TERMS IN NAME](#) in 1.5.4) or the name of a source group (see [DEFINING SOURCE GROUPS IN NAME](#) in 1.5.5) within the output request. If the calculation is constrained in this way, only material released from that source or source group is included for that field output. If no restriction is imposed then all sources are considered.

Similarly, when modelling particulate material, it is possible to request size-segregated output fields (for certain output quantities) based on a prescribed particle size distribution. This is currently supported for the following list of quantities: *Air Concentration*, *Mixing Ratio*, *Dry Deposition Rate*, *Wet Deposition Rate*, *Deposition Rate*, *# Particles*, *# Particles By Species*, *Mass*, *Mean Z*, *Sigma Z* and *Mean Travel Time*. Size-segregated output is requested by specifying a size distribution through the keyword [Particle Size Distribution](#) in the [Output Requirements - Fields](#) block, which should define the particle size-range bins that are to be used for the particle segregation.

i) Lagrangian quantities

Field name	Unit	Description
Air Concentration	(see text)	Air concentration computed from particles or puffs
Mixing Ratio	ppm / ppb	Volumetric mixing ratio relative to dry air
Dry Deposition Rate	(see text)	Dry deposition rate to the ground surface
Wet Deposition Rate	(see text)	Wet deposition rate to the ground surface
Deposition Rate	(see text)	Combined dry and wet deposition rates
# Particles	—	Number of particles by gridbox
# Particles By Species	—	Number of particles carrying a given species by gridbox
# Puffs	—	Number of puffs within computational domain
# Particle Steps	—	Particle steps computed from start of simulation
# Puff Steps	—	Puff steps computed from start of simulation
Mass	(see text)	Total mass of a given species that lies within the computational domain
Mean Z	(see text)	Mass-weighted mean plume height
Sigma Z	(see text)	Mass-weighted vertical plume spread
X Stats	(see text)	Mass-weighted plume spatial statistics returned as 10-element array: $\langle mass \rangle$, $\langle x \rangle$, $\langle y \rangle$, $\langle z \rangle$, $\langle x'x' \rangle$, $\langle y'y' \rangle$, $\langle z'z' \rangle$, $\langle x'y' \rangle$, $\langle y'z' \rangle$, $\langle z'x' \rangle$
Mean Travel Time	s	Mass-weighted mean travel time by gridbox
Puff Centres	(see text)	Air concentration calculated as if any puffs are particles located at the puff centres
Sigma C	(see text)	Estimate of standard deviation of concentration fluctuations
Area at risk	N/A	'Area at risk' (\sim distance into tail of distribution)
Photon Flux	$\text{m}^{-2} \text{s}^{-1}$	Photon flux for cloud gamma dose calculations
Adult Effective Cloud Gamma Dose	Sv s^{-1}	Adult effective cloud gamma dose
Adult Lung Cloud Gamma Dose	Gy s^{-1}	Adult lung cloud gamma dose
Adult Thyroid Cloud Gamma Dose	Gy s^{-1}	Adult thyroid cloud gamma dose
Adult Bone Surface Cloud Gamma Dose	Gy s^{-1}	Adult bone surface cloud gamma dose

ii) Eulerian quantities

Field name	Unit	Description
Eulerian Concentration	(see text)	Air concentration from field
Eulerian Total Deposition Rate	(see text)	Total deposition rate from field
Eulerian Dry Deposition Rate	(see text)	Dry deposition rate from field
Eulerian Wet Deposition Rate	(see text)	Wet deposition rate from field
E Mixing Ratio	ppm / ppb	Volumetric mixing ratio relative to dry air from field

iii) Combined quantities

Field name	Unit	Description
Concentration	(see text)	Air concentration combined from particles (or puffs) and field

Table 1.8: List of available NAME output fields arranged by their category

iv) Flow quantities

Field name	Unit	Description
Wind Speed	m s^{-1}	Mean wind speed
Wind Direction (degrees)	$^{\circ}$	Mean wind direction
Mean Flow U	m s^{-1}	u-component of mean flow
Mean Flow V	m s^{-1}	v-component of mean flow
Mean Flow W	m s^{-1}	w-component of mean flow
Temperature (C)	$^{\circ}\text{C}$	Air temperature in degrees Celsius
Temperature (K)	K	Air temperature in Kelvin
Potential Temperature (K)	K	Potential temperature
Pressure (Pa)	Pa	Atmospheric pressure
Sea Level Pressure (Pa)	Pa	Atmospheric pressure reduced to mean sea level
Density	kg m^{-3}	Air density
u-star	m s^{-1}	Friction velocity
Roughness Length	m	Surface roughness length
Sensible Heat Flux	W m^{-2}	Surface sensible heat flux
Pasquill Stability	—	Continuous form of Pasquill stability category (values in range 0 – 7)
Boundary Layer Depth	m	Boundary layer depth
Precipitation Rate (mm/hr)	mm hr^{-1}	Precipitation rate
Cloud Amount (oktas)	oktas	Total cloud cover in oktas
Relative Humidity (%)	%	Relative humidity
Specific Humidity	kg kg^{-1}	Specific humidity
Cloud Water (kg/kg)	kg kg^{-1}	Total (or dynamic) cloud liquid water content
Cloud Ice (kg/kg)	kg kg^{-1}	Total (or dynamic) cloud ice content
Topography	m	Topographic height of surface
Sigma VV	$\text{m}^2 \text{s}^{-2}$	Horizontal turbulent velocity variance
Mesoscale Sigma VV	$\text{m}^2 \text{s}^{-2}$	Horizontal velocity variance for unresolved mesoscale motions
Sigma WW	$\text{m}^2 \text{s}^{-2}$	Vertical turbulent velocity variance
Tau WW	s	Lagrangian timescale for vertical turbulence

Table 1.8: List of available NAME output fields arranged by their category (cont.)

iv) Flow quantities (cont.)

Field name	Unit	Description
3d Cloud (Fraction)	0 – 1	Three-dimensional total cloud fraction
Convective Cloud Base	m	Base of lowest convective cloud layer in m agl
Convective Cloud Top	m	Top of lowest convective cloud layer in m agl
Land Fraction	0 – 1	Land surface fraction
Land Use Fractions	0 – 1	Land use fractions (for 9 land-use categories as used in UM)
Canopy Water	kg m ⁻²	Canopy water content (for 5 canopy types as used in UM)
Canopy Height	m	Canopy height (for 5 canopy types as used in UM)
Leaf Area Index	0 – 1	Leaf area index (for 5 canopy types as used in UM)
Stomatal Conductance	m s ⁻¹	Stomatal conductance (for 5 canopy types as used in UM)
Soil Moisture	kg m ⁻²	Soil moisture content

v) Other quantities

Field name	Unit	Description
Progress (%)	%	Percentage of NAME run that has completed
Clock Time		Wall clock time
X	(see text)	x-coordinates of horizontal positions
Y	(see text)	y-coordinates of horizontal positions

Table 1.8: List of available NAME output fields arranged by their category (cont.)

i) Lagrangian quantities

Field name	Functional form	Need H-Coord?	Need Z-Coord?	Need Species?	Restrictable By Source?
Air Concentration	$f(x, y, z, t)$			■	■
Mixing Ratio	$f(x, y, z, t)$			■	■
Dry Deposition Rate	$f(x, y, t)$			■	■
Wet Deposition Rate	$f(x, y, t)$			■	■
Deposition Rate	$f(x, y, t)$			■	■
# Particles	$f(x, y, z, t)$				■
# Particles By Species	$f(x, y, z, t)$			■	■
# Puffs	$f(t)$				■
# Particle Steps	$f(t)$				■
# Puff Steps	$f(t)$				■
Mass	$f(t)$			■	■
Mean Z	$f(t)$		■	■	■
Sigma Z	$f(t)$		■	■	■
X Stats	$f(t, s)$	■	■	■	■
Mean Travel Time	$f(x, y, z, t)$			■	■
Puff Centres	$f(x, y, z, t)$			■	■
Sigma C	$f(x, y, z, t)$			■	■
Area at risk	$f(x, y, z, t)$			■	■
Photon Flux	$f(x, y, z, t)$			■	■
Adult Effective Cloud Gamma Dose	$f(x, y, z, t)$			■	■
Adult Lung Cloud Gamma Dose	$f(x, y, z, t)$			■	■
Adult Thyroid Cloud Gamma Dose	$f(x, y, z, t)$			■	■
Adult Bone Surface Cloud Gamma Dose	$f(x, y, z, t)$			■	■

Table 1.9: Characteristics and requirements of NAME output fields

ii) Eulerian quantities

Field name	Functional form	Need H-Coord?	Need Z-Coord?	Need Species?	Restrictable By Source?
Eulerian Concentration	$f(x, y, z, t)$			■	
Eulerian Total Deposition Rate	$f(x, y, t)$			■	
Eulerian Dry Deposition Rate	$f(x, y, t)$			■	
Eulerian Wet Deposition Rate	$f(x, y, t)$			■	
E Mixing Ratio	$f(x, y, z, t)$			■	

iii) Combined quantities

Field name	Functional form	Need H-Coord?	Need Z-Coord?	Need Species?	Restrictable By Source?
Concentration	$f(x, y, z, t)$			■	

iv) Flow quantities

Field name	Functional form	Need H-Coord?	Need Z-Coord?	Need Species?	Restrictable By Source?
Wind Speed	$f(x, y, z, t)$				
Wind Direction (degrees)	$f(x, y, z, t)$	■			
Mean Flow U	$f(x, y, z, t)$	■			
Mean Flow V	$f(x, y, z, t)$	■			
Mean Flow W	$f(x, y, z, t)$				
Temperature (C)	$f(x, y, z, t)$				
Temperature (K)	$f(x, y, z, t)$				
Potential Temperature (K)	$f(x, y, z, t)$				
Pressure (Pa)	$f(x, y, z, t)$				
Sea Level Pressure (Pa)	$f(x, y, z, t)$				
Density	$f(x, y, z, t)$				
u-star	$f(x, y, z, t)$				
Roughness Length	$f(x, y, z, t)$				
Sensible Heat Flux	$f(x, y, z, t)$				
Pasquill Stability	$f(x, y, z, t)$				
Boundary Layer Depth	$f(x, y, z, t)$				
Precipitation Rate (mm/hr)	$f(x, y, z, t)$				
Cloud Amount (oktas)	$f(x, y, z, t)$				

Table 1.9: Characteristics and requirements of NAME output fields (cont.)

iv) Flow quantities (cont.)

Field name	Functional form	Need H-Coord?	Need Z-Coord?	Need Species?	Restrictable By Source?
Relative Humidity (%)	$f(x, y, z, t)$				
Specific Humidity	$f(x, y, z, t)$				
Cloud Water (kg/kg)	$f(x, y, z, t)$				
Cloud Ice (kg/kg)	$f(x, y, z, t)$				
Topography	$f(x, y, z, t)$				
Sigma VV	$f(x, y, z, t)$	■			
Mesoscale Sigma VV	$f(x, y, z, t)$	■			
Sigma WW	$f(x, y, z, t)$				
Tau WW	$f(x, y, z, t)$				
3d Cloud (Fraction)	$f(x, y, z, t)$				
Convective Cloud Base	$f(x, y, z, t)$				
Convective Cloud Top	$f(x, y, z, t)$				
Land Fraction	$f(x, y, z, t)$				
Land Use Fractions	$f(x, y, z, t)$				
Canopy Water	$f(x, y, z, t)$				
Canopy Height	$f(x, y, z, t)$				
Leaf Area Index	$f(x, y, z, t)$				
Stomatal Conductance	$f(x, y, z, t)$				
Soil Moisture	$f(x, y, z, t)$				

v) Other quantities

Field name	Functional form	Need H-Coord?	Need Z-Coord?	Need Species?	Restrictable By Source?
Progress (%)	$f(t)$				
Clock Time	$f(t)$				
X	$f(x, y)$	■			
Y	$f(x, y)$	■			

Table 1.9: Characteristics and requirements of NAME output fields (cont.)

1.4.1.1 Spatial averaging

For Lagrangian output quantities computed from particles, there is an implicit spatial averaging applied in summing the contributions from all particles in each grid box. Such quantities should therefore be interpreted as grid box average values. In contrast, quantities computed from puffs and requests for flow information use the spot value at each grid point and, for these fields, spatial averaging over a grid box is not currently supported by NAME (although spatial averaging of puffs may be added in a future release).

It is also possible for the user to request explicit spatial averages or integrations for certain Lagrangian field outputs (viz. air concentration, deposition and particle number outputs). Ordinarily a horizontal grid and (with the exception of deposition fields) a vertical grid are specified for these quantities, however the following options are also available:

- If a horizontal grid is not supplied (i.e. the [H-Grid](#) variable is kept blank in the output request) then NAME calculates a horizontal integral evaluated over the entire computational domain.
- If a vertical grid could be supplied and is not (i.e. the [Z-Grid](#) variable is kept blank for a quantity other than a deposition field) then the field calculation depends on the value that is given for the [BL Average?](#) entry, as follows:
 - if given as [Yes](#), then an average value over the boundary layer depth is calculated;
 - if [No](#) (or left blank), then a vertical integral over the extent of the computational domain is calculated.

Spatial averaging is not supported for flow quantities, partly because it is often not obvious that such quantities are meaningful.

1.4.1.2 Temporal averaging

Time averages or time integrals can be requested when asking for output fields. This includes requests for meteorological or other flow information (although the calculation of an arithmetic mean might not be appropriate conceptually for some parameters such as wind direction). Time averaging or integrating is invoked using the variable [T Av Or Int](#) in the output request, which takes the possible values of [Av](#) to indicate that a time average is to be calculated, [Int](#) to indicate that a time integral is to be calculated, or [No](#) (or blank) to indicate no time averaging or integrating of the quantity. In the latter case, the instantaneous ‘spot’ value of the field is returned at each instant in time on the output time grid, whereas time averages or time integrals are computed over the time period *ending* at each time on the output time grid. For example, a request for an hourly average to be output on each hour would be interpreted as the time average over the one-hour period preceding each output time.

On requesting a time-averaged or time-integrated quantity, the user needs to specify the relevant time interval over which the averages or integrals are to be evaluated, and the number of instantaneous ‘contributions’ to be computed within that time period (i.e. the ‘density’ to be used for the averaging or integrating calculation). These parameters are supplied in the output request using the two variables **Av Time** (which should be formatted as a time interval, as defined in 1.2.2) and **# Av Times**, respectively. Both variables should be left blank for any requests that do not involve the time averaging or integrating of quantities. Note that it is most effective to calculate the instantaneous ‘contributions’ at synchronisation times (that is, when particles and puffs are synchronised within the model) and therefore the choice of **# Av Times** is usually informed by the **Sync Time** value (and vice-versa).

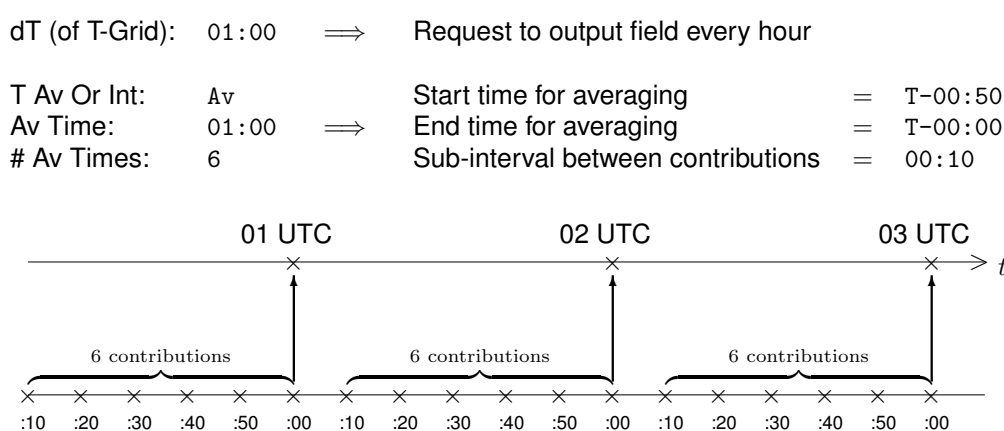


Figure 1.5: Schematic representation of time averaging or time integrating in NAME. Here an hourly average field, using contributions calculated at 10-minute intervals, is being requested every hour.

See Figure 1.5 for a schematic representation of the interpretation of time averages or integrals in NAME. When calculating a time average or time integral, the instantaneous quantity is first evaluated at the specified number of times within each averaging or integrating interval, and these separate contributions are then used to construct the required average or integral. The individual contributions are uniformly spaced within the time interval (e.g. if six contribution times are requested within a one hour averaging window then they will be equally spaced at 10 minute sub-intervals) but they are ‘right-justified’ with respect to the time window. In other words, the instantaneous quantity is effectively evaluated at the end of each sub-interval. As the number of contribution times increases, the computed average or integral based on this finite set of discrete values would be expected to approach the true computational value for a continuously varying quantity (although there are, of course, other discretisations in NAME which prevent this from happening in practice). However increasing the number of contribution times for calculating averages or integrals will also increase the computational cost of a NAME run and may not deliver significant benefit once the sub-interval of individual contributions becomes very small (relative to the time scale of the changes in the dispersing fields). In practice, it is most efficient to set the sub-interval to coincide

with the synchronisation time step (or a multiple thereof) that is used for advancing the individual particle calculations. Time averaging or integrating is often configured to process contributions at every synchronisation time, although less frequent contributions may be appropriate when [Sync Time](#) is small.

In practice, the number of averaging (or integrating) times is often selected to be 4, 6 or 12 per hour (which is equivalent to sub-intervals between the contribution times of 15 minutes, 10 minutes and 5 minutes, respectively), although a user may decide to use a different value depending on the nature of their run.

The choice of averaging (or integrating) time in a request is independent of the frequency at which output fields are written (i.e. the dT of the T-Grid). These two time intervals might often be chosen to be the same (e.g. an hourly-average value output every hour, or a daily average output every day), however it is equally possible to request, say, an hourly-average value output every three hours or a running 8-hour average output every hour.

1.4.1.3 Statistical processing

Additional processing of output fields can be requested in some situations, with NAME capable of calculating statistics over time or over an ensemble of realisations (e.g. based on the meteorology from different members of an ensemble NWP system).

Percentiles and exceedence probabilities can be calculated over a time period (e.g. the 98-th percentile of the hourly-average air concentration at a location over a one-year time period, or the proportion of exceedences of a threshold value for an air quality standard at that location). These statistical quantities are requested and processed in a similar way to time averages (at least conceptually) but also add an extra layer of complexity. When requesting percentiles or exceedence probabilities, the user needs to consider i) the time grid of the output (e.g. output frequency), ii) the time period over which each set of output statistics will be calculated, iii) the time step within that time period for which individual sample values will be evaluated, and iv) the quantity being computed as those individual samples (which may include time averages or integrals, spatial averaging, etc.). Figure 1.6 illustrates the main concepts in the specification of these statistical quantities.

The time grid of the output, as specified in the normal way by the [T-Grid](#) variable, gives the time or times at which the set of output statistics will be produced. For instance, one may require a single set of statistics covering a one-year period that are output just once at the end of that year, or one may be interested in weekly statistics from that year that are output at 52 separate times through the year. Each set of output statistics is then calculated using sample values of the underlying quantity evaluated at a uniform sampling rate over a preceding time period. The sampling period and sampling rate are specified in the output request using two time intervals [P Time](#) and [P dT](#), respectively. For example, a sample distribution of hourly data over a one-year period would be given by setting [P Time](#) to be [8760:00](#)³ and [P dT](#) to be [01:00](#). The underlying quantity at each

³As there are 8760 hours in a non-leap year, or alternatively the same time interval could be specified as 365d.

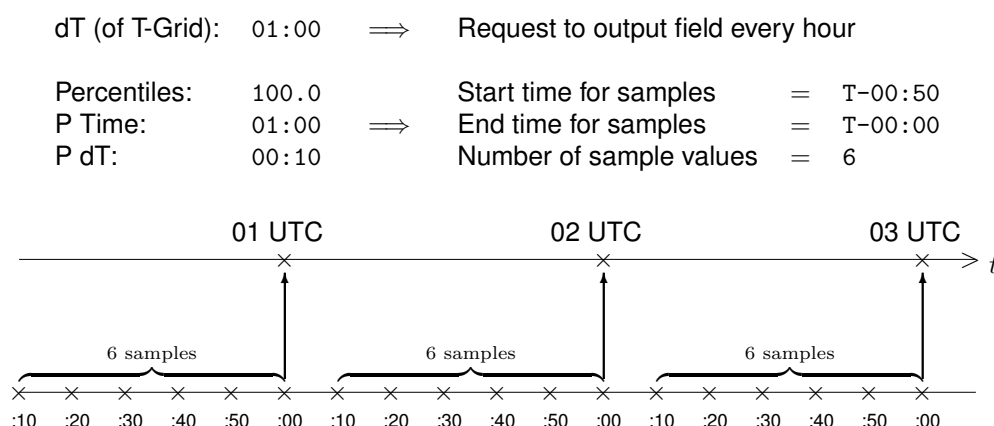


Figure 1.6: Schematic representation of the calculation of statistical quantities over time (in this example, the maximum 10-minute value in each one-hour period).

sample time is then subject to the spatial and temporal averaging prescribed for this field request (for instance, each individual sample could be a hourly average value).

Having constructed a sample distribution in the above manner, the user can request output statistics in one of two forms: percentiles or exceedence probabilities. Percentiles will be returned if a semicolon-separated list of the required percentiles is specified in [Percentiles](#); exceedence probabilities will be calculated if a semicolon-separated list of threshold values is specified in [Probabilities](#). Both options cannot be used together in the same output request, however percentiles and probabilities can both be requested within the same model run simply by using two separate requests.

For example, the minimum, median and maximum values from the sample distribution would be returned by a request in which [Percentiles](#) had the value [0.0 ; 50.0 ; 100.0](#), and the probabilities (i.e. sample proportion of exceedences) for air concentrations exceeding thresholds of 50 Bq m⁻³, 200 Bq m⁻³, 1000 Bq m⁻³ and 4000 Bq m⁻³ would be calculated when [Probabilities](#) had the value [50.0 ; 200.0 ; 1000.0 ; 4000.0](#).

Ensemble statistics of a quantity can be constructed over the ensemble of separate realisations produced in a multiple cases run (see [RUNNING MULTIPLE 'CASES'](#) in 1.2.7.5). The ensemble average (ensemble mean⁴) can be requested by setting the variable [Ensemble Av?](#) to [Yes](#). Alternatively, percentiles or exceedence probabilities over the ensemble of cases can be requested by setting the variable [Ensemble P?](#) to [Yes](#) in the output request, in conjunction with either a list of the required percentiles or of the threshold values to be used for calculating exceedences. Percentiles will be calculated when a semicolon-separated list of the required percentiles is specified in [Percentiles](#); exceedence probabilities will be calculated when a semicolon-separated list of threshold values is specified in [Probabilities](#). Both options cannot be used together in the same output request, however percentiles and probabilities can be requested within the same model run by using two

⁴If required, the ensemble median can be requested as the 50-th percentile of the ensemble distribution.

separate requests.

For example, the minimum, median and maximum values over the ensemble of cases would be returned by a request in which **Percentiles** had the value *0.0 ; 50.0 ; 100.0*, and the sample probabilities over an ensemble of cases for air concentrations exceeding thresholds of 50 Bq m⁻³, 200 Bq m⁻³, 1000 Bq m⁻³ and 4000 Bq m⁻³ would be calculated when **Probabilities** had the value *50.0 ; 200.0 ; 1000.0 ; 4000.0*.

Ensemble statistics are processed at the end of the NAME run after all the individual cases (or realisations) have completed, and ensemble results are then output in a separate set of files labelled as 'Case C'.

1.4.2 Requesting trajectory-based output (particle/puff information) from NAME

Information along individual particle (or puff) trajectories can be requested using an **Output Requirements - Sets of Particle/Puff Information** block in the set of main input files.

Each request for particle/puff information must be assigned a name, using the variable **Output Name**, which acts as a means of identifying that output requirement. The name is also used to label the output file(s) for that request. The user needs to specify a horizontal and vertical coordinate system, under **H-Coord** and **Z-Coord**, respectively, as the reference frame in which the particle positions are returned. A temporal grid, **T-Grid**, may be supplied to output trajectory information at a specified frequency, otherwise the information is output at every particle time step.

Supplementary information can be supplied to further qualify a request. For instance, trajectory information can be restricted to consider only particles or only puffs, or only certain ranges of particles or puffs (as identified by their unique reference index). It is also possible to restrict the output to particles/puffs released by a particular source (with all sources being considered if there is no restriction imposed here). The set of information that is generated by a request can be controlled, with options available to output particle/puff masses, details of the meteorology along a trajectory, plume-rise parameters, details of the dispersion scheme in use by that particle/puff, and details of the puff family (for puffs only).

There is also some user control of the format and structure of particle/puff information files, achieved through the **Output Format** keyword. Trajectory information from separate particles (or puffs) can be directed into separate output files. Similarly, trajectory information at separate times can be written to separate output files. This structuring can be useful as a large quantity of trajectory information can potentially be generated during a NAME run, which might be difficult to handle if all data were contained in a single output file.

See the >> **INPUT DOCUMENTATION** << for further details on requesting particle/puff information.

1.4.3 Requesting output of pdfs for concentration fluctuations

Probability distributions are a special type of output that can be requested when modelling short-duration concentration fluctuations, see [Short-term concentration fluctuations](#) in 1.7.12.

Fluctuations in the instantaneous concentration within a dispersing plume are created by the turbulent nature of diffusion in the atmospheric boundary layer, and they are especially significant at short ranges downwind of a compact source. These fluctuations are represented in NAME as a distribution of concentration values at each grid-point and at each time (rather than a single value as in the case of the mean concentration) and, as such, a probability distribution could be regarded as an extended form of field.

When modelling probability distributions for concentration fluctuations, output can be requested either as a collection of probabilities of exceeding defined concentration thresholds, or as a collection of percentiles of the concentration. The threshold values used for the calculation of exceedence probabilities and the percentiles are currently determined automatically by NAME, although an option to provide user-specified values will be added in the future. The selected values for the percentile calculations are listed in Table 1.10 (these are set as global parameters in the source code and could be modified, if necessary, with the NAME executable then rebuilt). The threshold values used for the exceedence probabilities are automatically selected according to the magnitude of concentration values at each grid point.

Percentile (%)	Percentile (%)	Percentile (%)
0.01	25.0	99.0
0.1	50.0	99.9
1.0	75.0	99.99
5.0	90.0	
10.0	95.0	

Table 1.10: Automatic percentile values calculated by fluctuations scheme.

The user can request probability distributions using an [Output Requirements - Pdfs](#) block in a similar way to a request for fields. The structure of an [Output Requirements - Pdfs](#) block is broadly the same as the corresponding block for fields, although there are fewer available options reflecting the fact that the fluctuations scheme is limited to a restricted range of modelling scenarios. At the same time, an additional keyword variable [Pdf Type](#) is included to allow the user to indicate the type of pdf information that is required. Two options are supported for the [Pdf Type](#) keyword: [1](#) is used to request output of a probability distribution as a collection of exceedence probabilities, whereas [2](#) is used to request output in the form of percentiles.

As with field requests, each request for pdf information is identified by a [Name](#), which is used to identify the output requirement internally within NAME and is also written in the output. All pdf requests should be given a unique name by the user, and although names can be largely arbitrary here, it is generally recommended to give a short, descriptive name to each request. The name of

the species for which the pdf is to be calculated is supplied in the variable [Species](#). The [Source](#) and [Source Group](#) variables can be defined to identify the source term to be considered. However, as the fluctuations scheme is currently designed to only treat a single source, then it is generally recommended to only consider one source in any model run requesting pdf information.

A horizontal grid, [H-Grid](#), vertical grid, [Z-Grid](#), and temporal grid, [T-Grid](#), should be provided in any request for a probability distribution. In particular, no implicit averaging is supported for pdf information (for instance it is not possible to ask for boundary-layer average results). Time averaging is currently not supported (although there are two keywords, [T Av Or Int](#) and [Average Time](#), which are intended to be analogous to their counterparts for requesting fields, but these are currently not used for probability distributions).



The modelling of concentration fluctuations is a research option in NAME, and has not been tested as extensively as other, more mature, aspects of the modelling system.

1.4.4 Format of NAME output files

There are two styles of output file used by NAME for text output of results. The first broad category of files is associated with the output of fields, which includes horizontal fields, vertical transects and time series (see [REQUESTING OUTPUT OF FIELDS FROM NAME](#) in 1.4.1). The second category covers the trajectory-based output of particle/puff information (see [REQUESTING TRAJECTORY-BASED OUTPUT \(PARTICLE/PUFF INFORMATION\) FROM NAME](#) in 1.4.2). The precise layout of an output file within each type is quite flexible and is controlled by the user. Many options are available for customising the information content and structure of the output file or files generated by NAME. However, some standard formatting rules are generally adopted for specific types of model output, and in particular, post-processing routines such as the generation of graphical products usually depend on the correct format styles being selected for the specific output request.

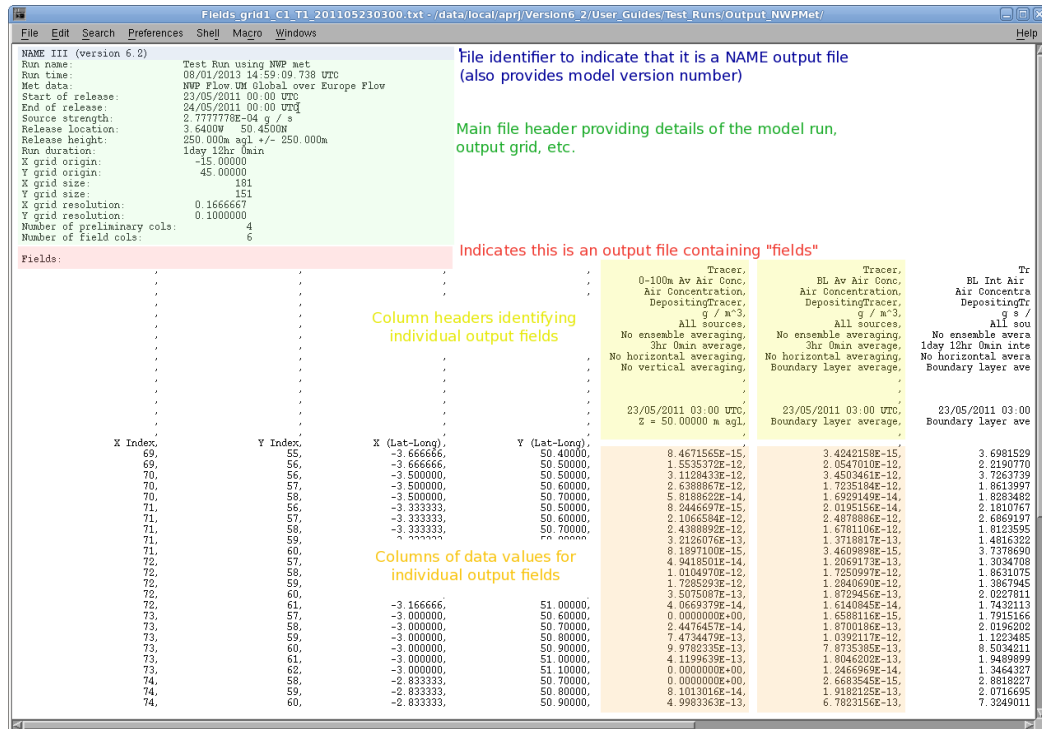
This section will describe the general layout of both styles of output file, and then consider some of the customisation and formatting options that can be specified to control the specific details of how NAME writes its output data. Required options for standard NAME output formats are also presented. See [» MD2/4: NAME III OUTPUT FILE FORMAT «](#) for further details about standard formats used when requesting output of fields and time series from NAME.



A new format for fields output is being developed – this is currently experimental and is invoked by setting the [Pre 6.5 Format?](#) keyword to [No](#) in the [Output Options](#) block.

1.4.4.1 Fields

The general structure of an output file containing fields information is illustrated in Figure 1.7, although the exact layout of any specific output file might vary somewhat from that shown.



File identifier to indicate that it is a NAME output file (also provides model version number)

Main file header providing details of the model run, output grid, etc.

Indicates this is an output file containing "fields"

Column headers identifying individual output fields

Columns of data values for individual output fields

Fields:		Tracer,		Tracer,		Tr	
		0-100m Av Air Conc,		BL Av Air Conc,		BL Int Air	
		Air Concentration,		Air Concentration,		Air Concentra	
		DepositingTracer,		DepositingTracer,		DepositingTr	
		g / m ³ ,		g / m ³ ,		g s /	
		All sources,		All sources,		All sou	
		No ensemble averaging,		No ensemble averaging,		No ensemble avera	
		3hr 0min average,		3hr 0min average,		1day 12hr 0min inte	
		No horizontal averaging,		No horizontal averaging,		No horizontal avera	
		No vertical averaging,		Boundary layer average,		Boundary layer ave	
		23/05/2011 03:00 UTC,		23/05/2011 03:00 UTC,		23/05/2011 03:00	
		Z = 50.00000 m agl,		Boundary layer average,		Boundary layer ave	
X Index,	Y Index,	X (Lat-Long),	Y (Lat-Long),				
69,	55,	-3.666666,	50.40000,	8.4671565E-15,	3.4242158E-15,	3.6981529	
69,	56,	-3.666666,	50.50000,	1.5535772E-12,	2.0547010E-10,	2.2190770	
70,	56,	-3.500000,	50.50000,	3.1128433E-12,	3.4503461E-12,	3.7263739	
70,	57,	-3.500000,	50.60000,	2.6388867E-12,	1.7235184E-12,	1.8613997	
70,	58,	-3.500000,	50.70000,	5.8198622E-14,	1.9292149E-14,	1.8283485	
71,	56,	-3.333333,	50.50000,	8.2446697E-15,	2.0195156E-14,	2.1810767	
71,	57,	-3.333333,	50.60000,	2.1066584E-12,	2.4878886E-12,	2.6869197	
71,	58,	-3.333333,	50.70000,	2.4388892E-12,	1.8781106E-12,	1.8123595	
71,	59,	-3.333333,	50.80000,	3.2126076E-13,	1.3718817E-13,	1.4816322	
71,	60,	-3.333333,	50.90000,	8.1897100E-15,	3.4609898E-15,	3.7378690	
72,	57,	-3.333333,	50.60000,	4.9418501E-14,	1.2069173E-13,	1.3034708	
72,	58,	-3.333333,	50.70000,	1.0104970E-12,	1.7250997E-12,	1.8631075	
72,	59,	-3.333333,	50.80000,	1.7285293E-12,	1.2840690E-12,	1.3867945	
72,	60,	-3.333333,	50.90000,	1.7285293E-12,	1.8729456E-13,	2.0227811	
73,	57,	-3.166666,	51.00000,	4.0669379E-14,	1.6140845E-14,	1.7432113	
73,	58,	-3.000000,	50.60000,	0.0000000E+00,	1.5588116E-15,	1.7915166	
73,	59,	-3.000000,	50.70000,	2.4476457E-14,	1.8700168E-13,	2.0196202	
73,	60,	-3.000000,	50.80000,	7.4734479E-13,	1.0392117E-12,	1.1223485	
73,	61,	-3.000000,	50.90000,	9.9782335E-13,	7.8735385E-13,	8.5034211	
73,	62,	-3.000000,	51.00000,	4.1139639E-13,	1.8046202E-13,	1.9489899	
74,	58,	-2.833333,	50.70000,	0.0000000E+00,	1.2465669E-14,	1.3464327	
74,	59,	-2.833333,	50.80000,	0.0000000E+00,	2.6683545E-15,	2.8818227	
74,	60,	-2.833333,	50.90000,	8.1013016E-14,	1.9182125E-13,	2.0716695	
				4.8983363E-13,	6.7823156E-13,	7.3489111	

Figure 1.7: Layout of a NAME output file containing fields information

The first entry indicates that this is a NAME output file, and also gives the version of the model that produced this file. The file identifier is followed by the main header block, which provides basic details about the model run (the run name, etc.) and the output grid for the fields contained in this file. The 'Fields:' indicator then shows that the file contains 'fields' output from a NAME run. Fields stored in the output file are then represented in a series of columns (with one field per column). These are preceded by a series of preliminary columns supplying coordinate or time information. Each field column has a header section that uniquely identifies the output field, followed by a columnar array prescribing the field data values at the corresponding coordinate points. The header elements contain the metadata describing field characteristics such as the quantity or the species being represented in the field. These metadata consist of a generic set of field properties, see Table 1.11, most of which are always defined for a field and written in the header section. However some coordinate values only have relevance when the corresponding variables appear in the [Across](#) attribute and these header entries are omitted from the output file when undefined (to clarify here, these header lines are actually *omitted* and not just left blank).

Output fields are grouped such that output field requests using the same group name are placed into the same output file or set of output files. The output group name for a field request is specified using the [Output Group](#) variable in the [Output Requirements - Fields](#) block. Each group name

Header element	Header description	Included when?
1	Species category	
2	Name	
3	Quantity	
4	Species	
5	Units	
6	Source/source group	
7	Particle size range*	
8	Ensemble averaging information	
9	Time averaging/integrating information	
10	Horizontal averaging/integrating information	
11	Vertical averaging/integrating information	
12	Probabilities and percentiles	
13	Probabilities and percentiles – over ensemble	
14	Probabilities and percentiles – over time	
15	Time	T ∈ Across
16	Location name	X, Y ∈ Across
17	X coordinate value	X ∈ Across
18	Y coordinate value	Y ∈ Across
19	Z coordinate value	Z ∈ Across
20	D coordinate value	D ∈ Across

Table 1.11: Master list of field column headers in a NAME output fields file. *Note that the particle size range is not currently included in the header unless the new experimental file format is selected.

should be a unique character string, and it is advisable to restrict the name to contain only alphanumeric and underscore characters (e.g. [Fields_grid1](#)) because the group name is also used as the basis for the prefix of the output filenames themselves. Fields assigned to the same output group will, of course, appear as multiple columns within the data files. It is also worth bearing in mind that multiple output columns might also be generated from a single field request depending on how the output is structured by the user. When multiple field requests share the same output group, care should be taken by the user to ensure that grids, etc. are consistent (NAME will report an appropriate error if it finds any inconsistencies which would prevent the required fields being written with the same file structure and grids).

The data layout of each requested output field is controlled using the two variables [Across](#) and [Separate File](#), each of which is a (possibly empty) string of characters indicating how the various field dimensions are to be treated when writing out the data. The dimensions that can be controlled in this way are: [X](#), [Y](#), [Z](#) for the three spatial dimensions, [T](#) for time, [S](#) for travel time, and [D](#) for data values (used when requesting statistical fields such as percentiles). The presence of any of these characters in the string supplied for the [Across](#) keyword indicates that the respective dimension is to be resolved *across* the output file or files using multiple columns (that is, the field at different values of that dimension are placed into different columns in the output file or files). These different

values of the dimension are then identified within the column header information. For example, the field values at different horizontal locations or for different vertical levels could be requested in separate columns by setting **Across** to **XY** or **Z**, respectively. Any dimension not specified in the **Across** keyword is resolved using separate lines (rows) in an output file, with the different values of the dimension given via the set of preliminary columns. This provides an alternative viewpoint for interpreting the layout that some users find easier to understand – any dimensions not listed as being *across* will instead appear *down* each output file.

The variable **Separate File** is used in a similar manner to **Across**, except that the instruction is to now write the field output at different values of the indicated dimension into separate output files. For instance, output at different locations can be split into separate files by including the character string **XY**, whereas field values at the individual times of a time grid can be written to separate files by adding the character **T**. In addition to the dimension characters, a further option exists for the **Separate File** keyword whereby the inclusion of **N** in the string of characters instructs NAME to start output afresh following a restart (see **RESTARTABLE NAME RUNS** in 1.2.7.1). When invoked with this option, any previous output files will be overwritten.

Note that for unstructured horizontal grids (i.e. locations), the dimensionality essentially reduces to one in the horizontal plane, and so to request different locations to be output in different columns (or to separate files) the user should ensure the pair **XY** is contained within the appropriate keyword value. For structured horizontal grids, where the horizontal dimensionality is properly two, it is possible to specify **X** and **Y** independently (which may be appropriate in some circumstances, for example, when requesting vertical transects in the x or y directions).

Although the broad structure of output field files is governed by the set of fields being requested in an output group and their associated settings for **Across** and **Separate File**, the user is able to control a number of aspects of the precise file layout using the **Output Format** keyword. As with the other keywords above, the **Output Format** variable consists of a string of characters indicating the required formatting options to be applied. The available options are listed in Table 1.12. Note that the **2** formatting option is only intended to be invoked where output files in the old NAME II format are required for legacy systems. **This is a deprecated format and should not be used for new applications** (in particular, the column header information is more restrictive in the NAME II format and this could potentially compromise field identification if the user is not careful).

Although the formatting variables introduced above provide the user with a high degree of flexibility in customising NAME output files to their specific requirements, there are also some standard output configurations which tend to be adopted in practice. Table 1.13 describes standard formats used for horizontal fields $f(x, y)$, vertical slices $f(x, z)$ or $f(y, z)$, and time series $f(t)$ outputs. Note that these standard formats must be used for the IDL plotting routines to work correctly.

Character String	Formatting Option
<i>I</i>	Include <u>I</u> ndices of the grid points as additional preliminary columns in each output file (these columns will appear in addition to the columns giving the actual grid point values themselves)
<i>A</i>	<u>A</u> lign the columns in each output file by padding with whitespace. This option is useful when the output is intended to be viewed directly by the user or when downstream use requires 'fixed-format' files. However, padding each file with spaces does come at the cost of increasing its file size and might not be necessary if, for instance, the output files are intended to be read as comma-separated files.
<i>Z</i>	Include grid points with <u>Z</u> ero values when writing each output file. By default, any grid point at which all fields are zero is usually omitted. As with the <i>A</i> option, including zero values in the output can increase the size of the output file, potentially by a large amount if the data fields are sparse.
<i>F</i>	Instructs NAME to <u>F</u> lush the (internal) file buffer after each write to the output file in order to keep the physical file up to date. For reasons of efficiency, Fortran programs tend to store any written data temporarily in an internal file buffer and then periodically write this to the physical file system in a series of chunks. The flush option should only be used with field requests that will generate small data files and where this functionality is really required. For example, output of the number of active particles or the percentage of the model run completed are two quantities where the flush option would be appropriate to use in order to properly monitor progress of the run in real time. The option should not be used for large output files as it will slow down the run.
<i>2</i>	Legacy option to emulate the output style of the previous NAME II modelling system.

Table 1.12: List of formatting options for field files output from NAME

Formatting Style	Grid Settings	Keyword Options	Headers Present (see Table 1.11)
Horizontal field	H-grid: structured regular Z-grid: optional T-grid: yes S-grid: no	<i>Separate File</i> = <i>T</i> <i>Across</i> = <i>[D]TZ</i> <i>Output Format</i> = <i>IA</i> <i>Output Group</i> = <i>Fields_*</i>	1 – 13, 14, 18, (19)
Vertical slice	H-grid: structured regular Z-grid: yes T-grid: optional S-grid: no	<i>Separate File</i> = <i>T</i> <i>Across</i> = <i>[D]TX</i> or <i>[D]TY</i> <i>Output Format</i> = <i>IA</i> [†] <i>Output Group</i> = <i>Fields_*</i> [†]	1 – 13, 14, 15, 16/17, (19)
Time series	H-grid: unstructured Z-grid: optional T-grid: yes S-grid: no	<i>Separate File</i> = <i>XY</i> or blank <i>Across</i> = <i>[D]XYZ</i> <i>Output Format</i> = <i>AZ</i> <i>Output Group</i> = <i>Time_series_*</i>	1 – 13, 15 – 18, (19)

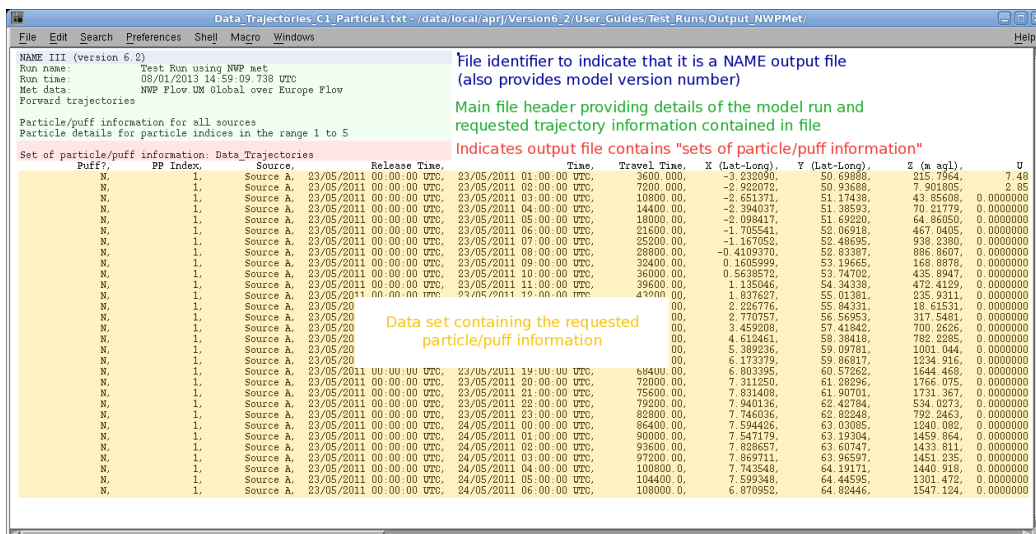
Table 1.13: Standard configurations for NAME output field files. [†]As there is no standard IDL script to plot vertical slices, any options can be specified for the output format and output group, but the preferred options listed are consistent with those used for horizontal fields.

1.4.4.2 Particle/puff information

The general structure of an output file containing particle/puff information is described in Figure 1.8, although once again the precise details of any specific file might vary from those illustrated. The first entry indicates that this is a NAME output file, and also gives the version of the model that produced this file. The file identifier is followed by the main header block, which provides basic details about the model run (the run name, etc.) and a summary of the trajectory information that has been requested by the user. The ‘Set of particle/puff information:’ indicator then shows that the file contains a set of particle/puff information (that is, trajectory-based output) from a NAME run. The particle/puff information is stored in the output file in a series of data records (rows) preceded by a single header row labelling the content of each variable that is present within a record.

One or more collections of particle/puff information can be requested from a NAME run by the user, with each request being identified by assigning a unique name for it in the **Output Name** variable of an **Output Requirements - Sets of Particle/Puff Information** block in the main input file. Each request for trajectory information is then written to a separate output file or set of output files. Each **Output Name** value should be a unique character string, and it is advisable to restrict the name to contain only alpha-numeric and underscore characters (e.g. *Data.Traj1*) because the name is used as the basis for the prefix of the output filenames themselves.

The structure of trajectory files is generally more rigid than that for output field files, because the underlying data essentially consists of single data records which are always written as individual lines in the output files. However some options are still available for the user to control the contents and format of these files. The list of parameters that are output along the particle/puff trajectories is controlled using the keywords *Met?*, *Mass?*, *Plume Rise?*, *Dispersion Scheme?* and *Puff Family?*,



File identifier to indicate that it is a NAME output file (also provides model version number)

Main file header providing details of the model run and requested trajectory information contained in file

Indicates output file contains "sets of particle/puff information"

Data set containing the requested particle/puff information

Figure 1.8: Layout of a NAME output file containing particle/puff trajectory information

which act as logical switches to include information on, respectively, the ambient meteorology at the particle position or puff centroid, the masses carried on the particle or puff, the state of variables involved in any plume rise calculation, the dispersion scheme being used by the particle or puff, and details of the puff family (for puffs only). Setting any of these keywords to **Yes** extends the collection of variables that are output for that request. Other features of the trajectory output can be controlled using the **Output Format** keyword, which consists of a string of characters indicating the required options to be applied (similar to the approach used for formatting fields output). The available options for trajectory outputs are listed in Table 1.14. When the **T** or **P** options are applied, trajectory information produced by the request is split across multiple output files. The individual files within such a set are identified by the relevant time or particle/puff index appearing in the filename.

Character String	Formatting Option
T	Indicates that the particle/puff information at separate <u>T</u> imes in a time grid should be written to separate output files (only for the case where an explicit time grid has been specified).
P	Indicates that the particle/puff information for separate <u>P</u> articles (or <u>P</u> uffs) should be written to separate output files.
F	Instructs NAME to <u>F</u> lush the (internal) file buffer after each sync time interval (main model time step) in order to keep the physical file up to date. For reasons of efficiency, Fortran programs tend to store any written data temporarily in an internal file buffer and then periodically write this to the physical file system in a series of chunks. The flush option should only be used where this functionality is really required, and use of the option for large output files can slow down a model run.

Table 1.14: List of formatting options for trajectory output from NAME

There are no 'standard' formatting options in use for trajectory outputs, except that **Met?** is usu-

ally set to [Yes](#) to include the meteorological information (and this is necessary if the IDL plotting routines are being used). Otherwise users have the freedom to select how particle/puff information is to be structured in the output files according to their particular needs – all the output can be contained within a single file or it can be split across particles or across times (but not across both particles and times). However one constraint is that [Output Name](#) must have the form [Data_*](#) for any trajectory information that is intended to be plotted using the IDL plot routine `plottraj.pro`. This is because the IDL routine matches on all files in the output folder following that specific naming convention. The trajectory plotting capabilities with IDL are rather basic and require certain consistency rules to be obeyed for trajectories (so that all trajectories to be plotted must cover the same time period). The intention is to develop a more flexible capability for plotting trajectory outputs in Python.

1.5 Source Terms in NAME

NAME has the capability to represent a wide variety of source types and source geometries and can handle situations ranging from a simple single point source to a complex collection of multiple sources. Each source might perhaps release a single species or a complex mixture of many species, while sources can also vary through time (e.g. a traffic emission cycle) or be dependent on the meteorology (e.g. biogenic emissions which might vary with air temperature, or disease transmission vectors such as midges which do not like wind and rain!).

This section will first consider how information on the species is used within NAME, and then examines the source terms themselves.

1.5.1 Defining species information in NAME

The dispersion of airborne material, and possible transformation or removal during transport, is partly determined by the underlying physical properties of the substance or substances involved. These substances, with their associated physical, chemical or biological characteristics, are referred to as *species*⁵ in NAME. Information on species is defined via a **Species** block in the set of main input files. A **Species** input block may contain multiple rows, with each line in the block specifying a separate species.

The user is able to prescribe various characteristics and properties of a species through the data supplied in its input block. For example, information on radiological characteristics, such as the half life of a radionuclide, or appropriate values to describe deposition behaviour can be provided in this way. Note that not all information about a species is necessarily stored with the species definition. For example, for chemicals or biological agents, certain properties and characteristics are sometimes 'hard-wired' in the model code in association with the species name.

In addition to the **Species** block defining the characteristics of the species themselves, there is also a **Species Uses** block (see [DEFINING SPECIES USE INFORMATION IN NAME](#) in 1.5.2) which specifies how those species are then used within NAME.



Sample species information is provided in the `Species.txt` file in the **Resources** folder. The user can extract the species of interest to their problem, editing any details as required. Alternatively, the user is free to create their own species having the desired characteristics.

1.5.1.1 Basic species properties

Three basic attributes that are associated with any species are its *name*, *category* and *material unit*, as defined using the keywords **Name**, **Category** and **Material Unit**, respectively.

⁵The word 'species' is used here in both a singular and plural sense, i.e. it can equally apply to a single substance or to a collection of multiple substances.

The user must assign a unique name to each species, which is then used to refer to this particular species in other aspects of the model set up (for example, when defining source terms that release the species, or requesting output quantities for that species). The name of the species also appears in the column header information written in the NAME output files. The species name should be a unique character string, and although names such as *Species 1*, *Species 2*, etc. are permitted by the naming convention, it is preferable to use descriptive names indicating the nature of the material (e.g. *Inert Tracer*) or an actual name for the substance (e.g. *Cs-137*, *Ammonia*).

The species category is a user-defined value providing a supplementary description of the nature of the species (e.g. the user may wish to clearly identify radioactive species by giving them the category *Radionuclide*). The category is read in with the species information and is written as part of the output file headers. However it has no other significance in the modelling, and may be left blank if such descriptive information is not required.

The material unit defines the unit in which the quantity of material is measured for the species. This is usually a physical unit, such as *g* (for gram) when the nature of the material is mass-based, or *Bq* (for becquerel) when the material is activity-based, although notional units may also be used here depending on the application. The material unit of a species can be regarded as providing a 'base unit' for that substance. By default, any release from a source term for this species will be given in the same unit, and similarly any output quantity requested for this species will be determined relative to that unit. However NAME does support some unit conversions such that it is possible to define source terms or to request outputs using different units to the base material unit. For instance, it would be possible to define a release of nuclear material in terabecquerels (TBq) and to request an integrated deposition field in kilobecquerels (kBq) per square metre. Further information on unit conversions in NAME is given in [▶ MD3/4: UNIT CONVERSIONS ◀](#).

1.5.1.2 Radiological characteristics

The keyword variable *Half Life* is used when modelling radionuclides to define the decay half-life for a radioactive species. For stable nuclides (that is, for species that are radiologically inert), the half-life value should be given as the text string *Stable*, which ensures that no decay in mass or activity will occur for that species. Otherwise the value supplied for a species half-life should be given in seconds. For further details on the modelling of radioactive decay in NAME, see [RADIOACTIVE DECAY](#) in 1.7.8.1.



Note that there is also a power law decay option available in NAME, which provides an alternative decay profile to half-life decay that may be appropriate for mixtures of radiological species. Power law decay is available via the *agent decay* functionality, see [MISCELLANEOUS SPECIES CHARACTERISTICS](#) in 1.5.1.6.

1.5.1.3 Advanced radiological options: decay chains and cloud gamma doses

NAME is capable of representing advanced radiological processes beyond a simple half-life mass loss due to decay of a radionuclide species. For instance, the activity associated with an (unstable) daughter product of a radioactive disintegration can, in turn, be considered, and this can be repeated for successive daughter products to enable treatment of decay chains. Currently only the primary decay route can be modelled within a chain (that is, for any species with more than one unstable daughter species, only one of the disintegration routes can be represented). However, if necessary, it is possible to model secondary decay routes involving daughter species having lower radiological significance by representing them using a duplicate collection of species within NAME.

Radioactive decay to a daughter species is specified using the two keywords [Daughter](#) and [Branching Ratio](#) in the definition of the parent species. Here the variable [Daughter](#) gives the name of the daughter product resulting from the decay of this parent, while [Branching Ratio](#) specifies the fraction of the activity of the parent which decays to this daughter (as a value between zero and one). Note that the daughter product must be defined in the set of species in its own right.

NAME can also model various types of cloud gamma dose experienced by receptors due to the irradiating effects of airborne material. The cloud gamma dose calculations for each radionuclide are based on a set of parameters defined in the input files using a [Cloud Gamma Parameters](#) block, see [DEFINING CLOUD GAMMA PARAMETERS FOR RADIOLOGICAL MODELLING](#) in 1.5.3. Each collection of cloud gamma parameters is identified by a unique name. The name is used to associate cloud gamma parameters with a radionuclide species by including the name under the [Set of Cloud Gamma Parameters](#) keyword in the species definition.

1.5.1.4 Dry deposition characteristics

The keywords [Surface resistance](#), [Deposition velocity](#), [Land use dependent dry dep](#) and [Mean aerosol diameter](#) are used to define species characteristics for dry deposition modelling. The [Surface resistance](#) variable allows the user to give a surface resistance value in s m^{-1} , from which a deposition velocity is calculated using a resistance analogy. Alternatively, a deposition velocity in m s^{-1} can be specified directly using the variable [Deposition velocity](#). [Land use dependent dry dep](#) is a logical variable which, when set to [Yes](#), denotes implementation of a land-use dependent dry deposition scheme. Finally, [Mean aerosol diameter](#) allows the user to specify a mean diameter in μm for modelling a non-sedimenting particle. No more than one of [Surface resistance](#), [Deposition velocity](#), [Land use dependent dry dep](#) or [Mean aerosol diameter](#) can be specified for an individual species (although it is permitted to use different options for different species, e.g., to define some species using a surface resistance and others with a deposition velocity).

Note that the land-use dependent dry deposition scheme cannot currently be used with puffs and it is only available for the following gaseous species: O₃, NO, SULPHUR-DIOXIDE, HYDROGEN, AMMONIA, CH₄, HCHO, PAN, NO₂, CO and HNO₃. For sedimenting particles, any value

specified for [Surface resistance](#), [Deposition velocity](#), [Land use dependent dry dep](#) or [Mean aerosol diameter](#) is ignored, and instead a dry deposition velocity is calculated by assuming zero surface resistance and a laminar sub-layer resistance obtained from the particle diameter. Here the particle diameter is specified through the source via [Particle diameter](#) or [Particle size distribution](#) (for sedimenting particles the diameter information is always specified with the source not the species). For non-sedimenting material, dry deposition only occurs if one of the variables [Surface resistance](#), [Deposition velocity](#), [Land use dependent dry dep](#) or [Mean aerosol diameter](#) is specified. Note that only one of these values can be specified for any individual species. Further details of the dry deposition options are given in [DEPOSITION AND SEDIMENTATION PROCESSES](#) in 1.7.7.

1.5.1.5 Wet deposition characteristics

The following keywords are available to define species characteristics for modelling wet deposition: [A rain - BC](#), [B rain - BC](#), [A snow - BC](#), [B snow - BC](#), [A rain - IC](#), [B rain - IC](#), [A snow - IC](#) and [B snow - IC](#). Given a pair of A and B coefficients, NAME calculates scavenging coefficients $\Lambda = Ar^B$, where r is the precipitation rate in mm hr^{-1} . Different A and B coefficients can be specified for below-cloud scavenging (BC) and in-cloud scavenging (IC) (often referred to as washout and rainout, respectively) and for snow and rain (i.e. the frozen and liquid precipitation components). The A and B coefficients must not be negative, although they can be set to zero. Note, in particular, that setting an A coefficient to zero will turn off wet deposition by rain / snow for that in-cloud / below cloud process. If none of the A and B coefficients are specified then no wet deposition occurs for that species. Further details of the wet deposition scheme can be found in [DEPOSITION AND SEDIMENTATION PROCESSES](#) in 1.7.7.

1.5.1.6 Miscellaneous species characteristics

There are some additional characteristics that can be defined for certain types of species.

Where the species represents a chemical element or compound, its *molecular weight* can be defined using the keyword [Molecular Weight](#). This information can then be accessed as necessary for chemistry calculations or for conversions between mass-based and volume-based concentration measures.

A simple scheme is implemented in NAME to represent mass loss (or deactivation) of a biological or chemical agent due to the action of ultra-violet radiation in sunlight. This is referred to as *UV decay* and forms part of the *agent decay* functionality in NAME. The scheme could be used, for example, to consider deactivation of FMD virus. When UV decay is being represented, the nominal loss rate (expressed as a percentage loss per hour, between 0% and 100%) can be defined using the keyword variable [UV Loss Rate](#). The nominal loss rate is defined assuming an overhead sun (that is, a solar zenith angle of zero) and this theoretical maximum value is modulated by the actual zenith angle. However no corrections are applied for any cloud cover (that is, the scheme assumes

clear-sky conditions).

A power law decay option is also available via the *agent decay* functionality. This uses the two keyword variables [Power Law Decay Exponent](#) and [Power Law Decay Delay](#), which define, respectively, the exponent of the power law decay profile and the particle travel time (or delay) before the power law decay starts. When using a power law decay, both of these values need to be defined and they should both be positive.

1.5.2 Defining species use information in NAME

The [Species](#) block is used to define the characteristics and properties of the species themselves, but gives no instruction on how those species are then used within a NAME run. For instance, whether the species is carried on model particles or is represented on a field, and whether a field is advected or static. Information on species use is supplied by a complementary [Species Uses](#) block.

All species that are used within a NAME run (on particles or on fields) must be first defined using one or more [Species](#) input blocks. A [Species Uses](#) block then describes how NAME uses those species. The block consists of a series of rows (one per species) and recognises the column keywords [Species](#), [On Particles?](#), [On Fields?](#), [Advect Field?](#) and [Particle Size Distribution For Fields](#). The name of the species is listed under the column header [Species](#) and the logical flags [On Particles?](#) and [On Fields?](#) indicate whether that species is to be modelled on computational particles and/or on an Eulerian field. For species on fields, the logical flag [Advect Field?](#) then indicates whether the field is to be advected by the Eulerian model within NAME (see [EULERIAN MODEL](#) in 1.7.11) or is a static field. If it is required that a species on fields should sediment, then this is indicated by naming a particle size distribution in the [Particle Size Distribution For Fields](#) column. Here the diameter range boundaries of the size distribution are used to group the material into particle size "bins" to be held on separate fields. Information on the particle density must also be included in the particle size distribution and (optionally) a representative diameter for each size range can also be included if needed.

Incidentally, the entries in the mass array on each computational particle are placed in a one-to-one correspondence with the list of species that are labelled for use "on particles" (i.e., for which the value of [On Particles?](#) has been set to [Yes](#) in the [Species Uses](#) block).

1.5.3 Defining cloud gamma parameters for radiological modelling

A [Cloud Gamma Parameters](#) block defines a collection of photon energies, and their associated cloud gamma parameters, relevant to a single radionuclide. These parameters are required for cloud gamma dose calculations. Multiple [Cloud Gamma Parameters](#) blocks can be present in the set of main input files if more than one radionuclide is to be considered. It is a 'named' type of input block, in which a supplementary name is associated with each separate instance of a block in order

to uniquely identify it. A cloud gamma parameters block is usually named after the radionuclide to which it relates (although other naming conventions could be followed provided that separate blocks are always identifiable by unique names).

A radioactive species may emit photons at multiple energy levels and although the photon energy itself is not used directly in the cloud gamma dose calculations other parameters are dependent on the energy of a photon. The information in a **Cloud Gamma Parameters** block defines these photon-energy dependent parameters necessary to calculate the cloud gamma dose attributable to radiation emitted by a radionuclide. Within a block, one line should be given per photon energy, with the keyword **Photon Energy** identifying the energy of a photon (or gamma-ray emission) in the unit MeV. The collection of parameters listed in Table 1.15 are then associated with each photon energy.

Keyword name	Parameter description
Photon Intensity	The relative frequency with which photons are released at this energy level (given as a value between 0 and 1). Note that these frequencies do not necessarily have to add up to 1 (as not all photon energies need to be considered).
Linear Attenuation Coefficient	The interaction probability per unit differential path length of a photon (m^{-1}). This factor describes how the radiation interacts with the matter through which it is passing.
B Build-up Factor a B Build-up Factor b	A pair of coefficients defining the Berger build-up factor, which describes the contribution to the photon flux at a receptor from the scattered photons.
Air kerma pu fluence	Air kerma per unit fluence (Gy m^2). This value is effectively the dose in air incident on a unit area.
Adult effective dose pu air kerma	Adult effective dose per unit air kerma (Sv Gy^{-1}) is a factor describing the whole body dose to an adult per unit dose in air.
Adult thyroid dose pu air kerma	Adult thyroid dose per unit air kerma (Gy Gy^{-1}).
Adult lung dose pu air kerma	Adult lung dose per unit air kerma (Gy Gy^{-1}).
Adult bone surface dose pu air kerma	Adult bone surface dose per unit air kerma (Gy Gy^{-1}).

Table 1.15: List of cloud gamma parameters associated with each photon energy.

See [MD4/2: IMPLEMENTATION OF CLOUD GAMMA MODELLING WITHIN NAME III](#) for additional details on defining cloud gamma parameters for dose calculations.

1.5.4 Defining source terms in NAME

Source terms are defined in NAME via one or more **Sources** blocks in the set of main input files. Each line in such an input block specifies a single source term. The user is able to prescribe many characteristics and properties of a source through the data supplied in its input block. For instance, information on the source location, geometry and spatial extent, the mass or activity that

is being released, or more advanced options such as emission temperature and exit velocity for stack emissions, can all be provided in this way.

1.5.4.1 Location and geometry

The geographical location and size of a source are specified by providing the spatial coordinates of the centroid and its extent about this central position in each of the x , y , and z directions. The basic source shapes supported by NAME are:

1. Cuboid (i.e. a rectangular-type geometry)
2. Ellipsoid (i.e. a circular-type geometry)
3. Cylindroid (i.e. a circular-type in the horizontal directions but rectangular-type in the vertical)

A source can be defined as zero-dimensional (i.e. a point source), as one-dimensional (a line source), as two-dimensional (an area source) or as three-dimensional (a volume source) according to its spatial extent in each of the x , y , and z directions. For example, a volume source would have a positive extent in all three directions, whereas a vertical line source would have a positive extent in the z direction only.

The location of the source centroid (X, Y, Z) can be specified using any horizontal and vertical coordinate systems that are supported by NAME and defined within the main set of input files. Alternatively, the geographical location of the source may have already been defined in a [Locations](#) block, in which case the name of that location may be used. Similarly, the spatial extent of the source (dX, dY, dZ) can be specified using these same coordinate systems or by using a metres-based measure. The latter option is useful, for instance, when a source location might be given in latitude-longitude coordinates but its size is known in metres (e.g. an industrial stack emission).

There are two options available for specifying the spatial distribution (or spatial density) of the release about its centroid:

1. Uniform (also referred to as a 'top hat' distribution) to give a flat spread throughout the source region, or
2. Gaussian (with the spread of the Gaussian calculated to give the same mass-weighted spread as the corresponding uniform distribution) to give a peak at the centroid that falls off with increasing distance.

It is also possible to specify an offset angle allowing a horizontal rotation of a source with respect to the defined (x, y) coordinate frame. Rotated sources could be useful, for instance, in defining emissions from a road traffic network (where each road section might be given as a separate angled source). This last example also illustrates the ability to construct complex source geometries by compounding the basic source types listed above.

1.5.4.2 Defining release characteristics for instantaneous, finite-duration and continuous releases

The temporal extent of a release can be specified by prescribing its start time and end time. Three regimes can be identified:

1. Instantaneous release, in which the start time and end time are the same finite time (i.e. the entire release occurs at a single moment in time)
2. Finite-duration release, in which the start time and end time are both finite with the end time occurring later than the start time
3. Continuous release, in which the start time is in the infinite past and/or the end time is in the infinite future

An alternative specification of the temporal extent can be given in terms of the release start time and the release duration, in which case the three regimes identified above are associated with release durations of zero, a finite time interval, and an infinite interval, respectively.

By default, a release over a finite or infinite time period will occur at a uniform rate (see later for the ability to specify time-varying source terms).

The specification of the mass or activity to be released by a source, as well as the number of particles to be used to model that source, depends on the release type. For an instantaneous release, the user should supply the total mass and the total number of particles to be released from that source. For a finite-duration release, the user can provide either a release rate per unit time or a total release from that source (from which a release rate is then calculated) for both the mass and the number of particles. For a continuous release, the user must only provide a release rate for the mass and number of particles (because the total release is, in principle, undefined for an infinite source). Note that the prescription of particle numbers is ignored here if the puff dispersion scheme is being used.

When defining the number of particles to be released (or the rate of release of particles), the **# Particles** keyword in the **Sources** block prescribes a *lower limit* on the number of particles released or particle release rate. The value should be given as a positive integer n when defining a total number of particles to release, or otherwise as a release rate in the form n / time , where supported time units for rates are: per second ('sec' or 's'), per minute ('min'), per hour ('hr'), and per day ('day'). The value supplied for **# Particles** usually defines the actual number of particles released by a source, but more generally it gives a lower bound for the number. This is because the mass or activity that can be allocated to any single particle might be restricted, see [IMPOSING LIMITS ON THE PARTICLE MASS](#) in 1.5.4.9, which imposes a further control on the particle release that can force the creation of additional particles when needed. This extra functionality is most frequently invoked when the NAME chemistry scheme is being used.

Note also that particle releases are constrained by the total number of particles allowed in the simulation, which is specified in the value [Max # Particles](#) in the [Sets of Dispersion Options](#) block. The user therefore needs to ensure that a sufficiently large limit is set on the global number of particles to accommodate their intended release terms (and any particle splitting that might occur); see [RUNNING NAME WITH PARTICLES](#) in 1.7.2 for further details.

When defining the mass to release, one or more species can be released from any source term. Each released species is prescribed using an instance of the [Source Strength](#) variable in a [Sources](#) block. So, for example, if a source were to release four species then there would be four copies of the [Source Strength](#) variable – one for each species. The associated value of this variable should take the following form:

Species_Name Amount *Material_Unit* [/ *Time_Unit*]

where *Species_Name* refers to the name of a species previously defined in a [Species](#) block, *Amount* refers to the mass or activity to be released for that species, *Material_Unit* refers to the material unit for that species (again defined in the [Species](#) block), and optionally a time unit, *Time_Unit*, which indicates that the release amount is a rate per unit time. Supported time units for rates are the same as for particle numbers (as listed above). As an example, a release from a source of one tera-Becquerel of Cs-137 per hour would be represented by the value Cs-137 1.0E12 Bq / hr.

1.5.4.3 Time-varying sources

When modelling an instantaneous source, all the material is released into the atmosphere on a collection of particles (or puffs) at a single moment in time. For a non-instantaneous source, the material is released gradually as a succession of particles over the specified duration of that source. Ordinarily the mass or activity is released at a steady constant rate over that period, however the default behaviour can be modified to allow for source strengths that vary with time. The variation may be a direct function of time, as described in this section, or may be an indirect consequence of changes in other variables through time such as the evolving state of the meteorology (see [MET-DEPENDENT SOURCES](#) in 1.5.4.6).

An explicit temporal modulation of source strengths can be defined via a [Source Time Dependency](#) block, which can then be applied to vary the emissions from a source term. The approach allows the user to specify a collection of scaling factors for the emissions alongside the corresponding time periods over which those factors should be applied. Up to five criteria can be applied to define the time periods, with support available for ‘wildcards’ in dates and times. For example, the user would be able to select daytime hours (from 09:00 to 17:00) for workdays (from Monday to Friday) in the winter months (from December to February). The method is useful for representing sources that vary with time on a regular or semi-regular basis (e.g., a daily or weekly traffic-cycle modulation of vehicle emissions, or a seasonal variation in power plant activity). See the [INPUT DOCUMENTATION](#) [◀](#) for further details.

1.5.4.4 Gridded met-dependent sources (mineral dust and sea-salt schemes)

There are two special types of source in NAME for *mineral dust* and *sea salt*. Both of these source types have two characteristic features: that their emission strengths are dependent on the meteorological conditions (and also on land surface properties) and that they are gridded sources (that is, emissions are defined relative to an underlying horizontal grid structure). Together these properties allow the spatial variability of source strength to be considered.

Dust and sea salt sources are sedimenting particle releases and have an associated particle size distribution. Special rules apply to these distributions for dust and sea-salt, see [PARTICLE SIZE DISTRIBUTIONS](#) in 1.2.4 for full details. In the case of a dust source, a unique particle diameter is associated with each size bin (the logarithmic mid-point is used here). These size bins are based on the input soil fraction size ranges (from ancillary files). For a sea-salt source, there are two fixed particle size ranges, and a random diameter is determined using a polynomial distribution within each of these diameter ranges. For both source types, the specified cumulative fraction in the input file is not used, but instead the mass released in each size range is determined directly from the meteorological and land surface data. Source strength for dust emissions is related to the friction velocity, whereas sea salt emissions are determined by the wind speed at 10 metres. In each case, emissions are calculated separately for each grid cell and particle size range. Both source types are surface releases.

Keyword variable	Value
Met-dependent Source Type	<i>Dust</i> (for a dust source) <i>Sea Salt</i> (for a sea salt source)
H-Grid	<i>must be specified</i>
Z-Coord	<i>m agl</i>
dZ-Metres	<i>Yes</i>
Z	<i>0.0</i>
dZ	<i>0.0</i>
Source Strength	<i>PM10_MINERAL ? g / s</i> (for dust) <i>PM10_SEA_SALT ? g / s</i> (for sea salt)
Particle Density	<i>must be specified</i>
Particle Size Distribution	<i>must be specified</i>

Table 1.16: Specific settings in the [Sources](#) block for dust and sea salt releases

Table 1.16 gives specific variable settings for the [Sources](#) block when representing mineral dust and sea salt releases in NAME. In particular, a dust or sea salt source must be explicitly requested by setting the keyword [Met-dependent Source Type](#) in the [Sources](#) block as *Dust* or *Sea Salt*, respectively. The name of a horizontal grid must also be supplied in the variable [H-Grid](#) on which the dust or sea salt releases will be calculated, but note that a [Z-Grid](#) should not be specified. There are pre-defined species **PM10_MINERAL** and **PM10_SEA_SALT** in the various species files that are designed to be used with mineral dust and sea salt sources, respectively, and here the [Material Unit](#)

variable in the [Species](#) block must be given as [g](#). Furthermore, only one species can be used for a dust or sea salt source. Both mineral dust and sea salt releases require the use of ancillary files in addition to the input meteorological data. Dust requires that surface and soil attributes are defined in the collection of flow module instances, whereas sea salt only requires surface attributes to be defined. Further details on using ancillaries within NAME are given in [USING LAND SURFACE FIELDS FROM NWP ANCILLARIES](#) in 1.6.2.1. Time dependencies cannot be specified, and plume rise and the fixed met options are not applicable for dust and sea-salt sources.

1.5.4.5 Estimating source mass flux for volcanic eruptions

Another type of met-dependent source represented in NAME is used for modelling the mass flux of volcanic plumes. Estimates of the source mass flux for volcanic eruptions are usually inferred indirectly from the rise height of the plume or eruption column. Empirical relationships, such as those proposed by Mastin et al. [2009] and [Sparks et al., 1997, §5.2], relate the rise height to the mass emission rate but do not take explicit account of the state of the atmosphere at the time of the eruption. A one-dimensional volcanic plume-rise model has been added to NAME that estimates iteratively the source mass flux for a given rise height. The model incorporates the effects of the ambient wind, moisture and temperature – using realistic profiles taken from the Unified Model – includes phase changes between liquid water and water vapour and accounts for bulk density and specific heat capacity that depend on the relative contributions of the different phases. Various parameters of the model such as source temperature, exit velocity, initial gas mass fraction and moisture content can be specified as inputs; default values exist for all such quantities which enables the model to run even if they are not specified by the user.

The model is activated for a source by setting the [Met-Dependent Source Type](#) keyword to [Iterative Plume Rise Model](#) in the [Sources](#) block. In order to run the model it is necessary to specify a vertical grid for the keyword variable [Z-Grid Iterative Plume Rise](#). It is also necessary to specify the target rise height in [Plume Rise Height](#) and the height of the source vent under [Summit Elevation](#). Additional (optional) inputs to the scheme are the initial gas mass fraction of the plume, [Dry Gas Mass Fraction](#), the initial moisture content, [Water Vapour Mass Fraction](#), and the fraction of fine ash retained in the distal plume, [Distal Fine Ash Fraction](#). Output is obtained by requesting a field with the [Quantity](#) keyword given as [Revised Source Strength](#). Equivalent results using Mastin's empirical formula can be requested similarly using [Original Source Strength](#). It is possible to request time-averaged or time-integrated values of either quantity.

The plume-rise model can also be used to estimate the source mass flux of (non-volcanic) gas-dominated plumes. Further details of this and similar models can be found in, e.g., Devenish [2013], Woodhouse et al. [2013] and Mastin [2014].

1.5.4.6 Other met-dependent sources

Apart from the gridded met-dependent sources of dust and sea-salt, and the special iterative plume-rise model for estimating source mass flux, several other source types also have a dependency on the meteorological conditions. These met dependencies are configured on the species name, and currently three such dependencies are supported by NAME in this way.

Species: MIDGE midges are released according to a Midge Dynamics Equation that determines the midge population becoming airborne as a function of the time of year, ambient air temperature, 10-metre wind speed and precipitation.

Species: C10H16bio biogenic α -pinene emissions have a temperature dependence.

Species: RESUSPENDED_ASH emissions of resuspended volcanic ash are dependent on the friction velocity and absence of precipitation.

In principle, a user could include their own met-dependency in a source term in a similar way.

1.5.4.7 Buoyant or momentum-driven releases

NAME has a plume rise scheme for modelling buoyancy or momentum driven sources. The scheme is invoked by setting the appropriate source characteristics using the variables [Plume Rise?](#), [Temperature](#) and either [Volume Flow Rate](#) or [Flow Velocity](#). The [Plume Rise?](#) variable has a logical value ([Yes](#) or [No](#)), which acts as a switch to turn the plume rise scheme on for any given source. [Temperature](#) is the stack emission temperature (in Kelvin)⁶, [Volume Flow Rate](#) is the volume flow rate (in $\text{m}^3 \text{s}^{-1}$) and [Flow Velocity](#) is the exit velocity (in m s^{-1}) of the release from the source, where only one of [Volume Flow Rate](#) or [Flow Velocity](#) should be specified. Note that these emission variables are ignored if the plume rise scheme is not invoked for the source (i.e. the [Plume Rise?](#) variable is set to [No](#) or is absent from the source specification). Further details on using the NAME plume rise scheme can be found in [PLUME RISE SCHEME FOR BUOYANT AND MOMENTUM-DRIVEN RELEASES](#) in 1.7.6.

1.5.4.8 Sedimenting releases

Gravitational settling of heavy particles (or sedimentation, as it is sometimes called) is invoked for a source using the variables [Particle Density](#), [Particle Shape](#), [Particle Diameter](#) and [Particle Size Distribution](#). When a [Particle Shape](#) is given, the user must also specify the [Sedimentation Scheme](#) that they wish to use to calculate the sedimentation of non-spherical volcanic ash particles, via the [Sources](#) block. When [Particle Shape](#) is not defined, particles are assumed to be spherical and the White [1974] sedimentation scheme is used.

⁶To convert an emission temperature given in $^{\circ}\text{C}$ to K, add 273.15 to the $^{\circ}\text{C}$ value.

A **Particle Density** (in kg m^{-3}), and either **Particle Diameter** or **Particle Size Distribution** should be specified for a sedimenting particle release. When a **Particle Diameter** (in μm) is supplied, all particles released from that source will have the prescribed diameter (i.e., the source will release particles of one single size). If a range of particles of varying sizes is required, the user should instead provide a **Particle Size Distribution**, which should refer to a named particle size distribution defined in a **Particle Size Distribution** block in the input file. Additionally, **Particle Density** and **Particle Shape** values can be defined as a function of particle size range via the **Particle Size Distribution** block. Details of how to define a particle size distribution can be found in **DEFINING PARTICLE SIZE DISTRIBUTIONS** in 1.2.4. For sedimenting particles, the size associated with the particle for sedimentation is also used for other purposes (such as dry deposition). Gravitational settling of particles is turned off for a source by leaving all of the variables **Particle Density**, **Particle Shape**, **Particle Diameter** and **Particle Size Distribution** blank.

NAME calculates a sedimentation velocity for each particle from its density, shape and diameter (which may have been specified directly or obtained from a prescribed particle size distribution). See **DEPOSITION AND SEDIMENTATION PROCESSES** in 1.7.7 for details of the sedimentation velocity calculations. Species or material with different densities, diameters, shapes or particle size distributions must be modelled as different sources, as should sedimenting and non-sedimentating species. Two specific types of particulate sources are dust and sea-salt sources, which are modelled as sedimenting releases by specifying a **Particle Density** and a **Particle Size Distribution** (but not a **Particle Diameter**). Further details of using the dust and sea-salt schemes is given in **GRIDDED MET-DEPENDENT SOURCES (MINERAL DUST AND SEA-SALT SCHEMES)** in 1.5.4.4.

1.5.4.9 Imposing limits on the particle mass

The user can apply limits to the amount of mass carried on individual particles on a species-by-species basis. For example, it would be possible to constrain, say, the mass of ozone that can be held by any single particle when modelling ozone on particles. If a quantity of ozone to be represented were then to exceed that limit, then additional particles would be brought into use to accommodate the mass restriction, either by releasing more particles (in the case of primary emissions) or through particle splitting (for secondary mass created through the chemistry scheme). This ability to constrain particle mass is usually invoked when modelling distributed releases involving a large number of sources, such as an air quality type of run, and helps to ensure a balanced distribution of mass on particles from different sources. In particular, it can help to avoid the situation where a 'small' source might release many 'light' particles while a 'large' source releases lots of 'heavy' particles, that can result in a sub-optimal mixture of particle mass sizes.

A particle mass limit can be associated with any of the species defined in the **Species** input block (which are in a one-to-one correspondence with the elements of the mass array for each particle). When two or more mass limits apply for any given particle, then the most restrictive constraint is applied (so that no species is allowed to exceed its mass limit). If no mass limit is associated with a

species, then the particle mass remains unrestricted for that species.

Particle mass limits are specified using a **Particle Mass Limits** block in the set of input files. This defines mass limits by species. Where the user requires a mass limit to be imposed on particles for a given species, then that species name should be listed under the **Species Name** keyword, and the corresponding upper limit on mass for that species provided under the **Particle Mass Limit** variable. Here the mass limit is expressed in the material unit of that species (as stated in the species definition). Mass limits for different species appear as separate lines in the block. An example of a **Particle Mass Limits** block is given in Table 1.17, in which mass limits are defined for two species, *SULPHUR-DIOXIDE* and *AMMONIA*.

Particle Mass Limits:	
Species Name,	Particle Mass Limit
SULPHUR-DIOXIDE,	20000.0
AMMONIA,	15000.0

Table 1.17: Example of setting limits on particle mass

1.5.5 Defining source groups in NAME

Sources can be grouped together in NAME to create *source groups*. This functionality is especially useful for attribution studies. For example, when conducting an air quality study, a user may decide to group all traffic sources together so that the combined traffic emissions can be assessed independently of other types of source. Furthermore, a source can be assigned to more than one source group, so that a UK traffic emission might be placed, say, into one group comprising of all UK sources as well as a second group of all traffic sources. The structure of any source groupings is very flexible and is controlled by the user.

Each source group is identified by a unique name supplied by the user, and although the naming convention here is arbitrary, it is generally recommended to use a short descriptive name. Source groups are defined as part of the information contained within a **Sources** block by using the **Source Groups** keyword. When a source is to be assigned to one (or more) source groups, the **Source Groups** keyword should contain a semicolon-separated list of the source group(s) to which that source belongs. When a source has no association with any source group, then the variable should be left blank. Source groups can then be used to restrict the calculation of output fields to only consider material originating from sources within a specified group (similar to the manner in which outputs can be restricted to a particular source). Restriction of output to a particular source or source group is specified as part of the **Output Requirements - Fields** block, see **REQUESTING OUTPUT OF FIELDS FROM NAME** in 1.4.1. Specifically, the required source or source group is set using the **Source** or **Source Group** keyword, respectively, within the relevant field request. No source restrictions are applied if both these values are blank.

1.5.6 Controlling particle numbers in a NAME run

NAME, as a Lagrangian model, operates by releasing large numbers of particles into its modelled atmosphere and then tracking their subsequent evolution. The number of active particles at any time is determined by various factors but particle numbers are ultimately governed by the balance between the release of new particles from sources (or the creation of additional particles through splitting of massive particles) and the removal of particles from the computational domain. Many aspects of the model set up can influence particle numbers, such as the number of sources being modelled and their individual release characteristics, the extent of the geographical domain being considered, and the length of the simulation period.

The use of more particles when representing the sources will, in principle, provide more reliable and robust results (especially towards the edges of plumes where particle number density is low). However, use of more particles in the simulation comes with a larger computational overhead, both in terms of the run time required to complete the simulation and in terms of the extra memory requirements (RAM) placed on the computing resources. In practice, the user needs to balance their requirements for model results against the available computational resource and time constraints. The key concept here is to obtain a well-balanced modelling set up to ensure that the available resources are utilised to best effect. For example, that different sources are treated according to their relative importance, and that the resolution of any output grid is consistent with the intrinsic level of information resolved in the simulation (which also depends on the 'scale' of the dispersion problem being considered, the meteorological input data, etc.). Achieving a balanced set up improves with experience and may require some initial experimentation, especially when considering complex simulations such as air quality scenarios which would usually involve a very large number of source terms and include treatment of the chemistry. Setting up NAME appropriately for a particular modelling scenario typically requires a certain degree of expertise in using NAME and understanding its limitations.

The user is able to exercise various levels of control over particle numbers, regulating release rates at the level of individual sources, for particular species, and at a global level. Basic release rates are specified on a per-source basis, see [DEFINING RELEASE CHARACTERISTICS FOR INSTANTANEOUS, FINITE-DURATION AND CONTINUOUS RELEASES](#) in 1.5.4.2, and should be chosen appropriately for the application. When modelling only a single source, or a small set of sources, a relatively high release rate can be considered. This could typically be of the order of 1,000 – 10,000 particles per hour for a continuous release, depending on the application. When modelling a large collection of sources in an air quality type of run, resource constraints impose far more modest release rates of perhaps 5 – 10 particles per hour. As a general principle, the particle release rate should scale with the significance of a source, so that any larger, more important sources should tend to release more particles than the smaller minor ones. This approach ensures that particles from different sources will all carry broadly the same amount of mass, and so avoid the problem of having 'cannonballs

and snowflakes’! The equalisation of particle masses across different sources is also aided by the ability to limit the amount of mass that can be assigned to any individual particle on a per-species basis, see [IMPOSING LIMITS ON THE PARTICLE MASS](#) in 1.5.4.9. This again helps to avoid the ‘cannon-balls and snowflakes’ problem as it ensures that, when defined, species-based limits are applied to all particles. Finally, there is a global level of control on particle numbers in which all particle release rates can be universally reduced as the global particle limit is approached, see [SPECIFYING DISPERSION MODELLING OPTIONS](#) in 1.7.1. This is designed to reduce the risk of the model completely running out of available particles to release at any point during the simulation.

If the maximum particle limit is reached, however, then all subsequent particle releases (and splitting of any massive particles) are stopped until existing particles are freed up and become available for re-use. Reaching the particle limit is an undesirable situation, as it implies that some sources or aspects of the particle splitting are not being treated properly in the model run (and warning messages will be produced to highlight such issues). NAME will, however, continue running normally in all other aspects and will continue to produce output as requested until particle releases can be resumed again. The model is deliberately designed to continue running when the particle limit has been reached in order to accommodate such an event happening on a rare occasion (for example, an extreme build-up of pollution particles during an extended anticyclonic period). However, if warning messages about the particle limit being reached are being generated on a regular basis, this indicates that the model set up is not consistent with the existing particle limit and the user should consider raising the particle limit (if run time and memory constraints will permit) or otherwise modify their configuration to reduce particle numbers in their model run.

1.6 Meteorology in NAME

As with any dispersion modelling system, NAME requires a source of input meteorology in order to provide flow information to drive the model simulations. NAME supports a variety of sources for its meteorological data, known technically as *met module types*, but the two primary types are **single-site meteorology** and **NWP meteorology**. Single-site meteorology typically consists of hourly series of observed meteorological parameters (such as wind speed, wind direction, temperature and cloud amount) recorded at a weather station, while NWP meteorology refers to the gridded data sets of weather variables generated by a **Numerical Weather Prediction** model.

The role of a *met module* in NAME is to read, organise and store meteorological data of some kind, for subsequent use by an associated *flow module*. One or more *instances* of a met module can be specified for any given met module type.⁷ Multiple instances are most commonly associated with the NWP met module (where met data from different scale NWP models might be used in a nested sense, or where met data from an NWP model might be split across multiple regions), although it is perfectly valid to define multiple instances for other met types as well.

The information in a met module is subsequently processed by a *flow module*. The purpose of a flow module is to provide met and flow information to other components in NAME, as and when requested by those model components. For example, a flow module could be tasked with supplying mean flow and turbulence quantities to the particle advection scheme, or temperature, pressure and humidity values to the chemistry scheme. To meet any requests, a flow module might perform simple processing on the input meteorological data stored within the met module (e.g. interpolating the available variables in space and time) or it can involve more advanced processing such as constructing vertical profiles or estimating turbulence information.

As with the met modules, multiple instances of a flow module can be defined for any flow module type. Flow modules are usually associated with met modules of the same type (for instance, single-site flow with single-site met, or NWP flow with NWP met), although this does not necessarily have to be the case. For example, there are some ‘local’ flow modules for modelling the flow deformation effects of an isolated building and for modelling small-scale terrain effects, which take their input meteorology from a large-scale met module.

The focus of the discussion here in the user guide will be on some of the broader aspects of setting up and using sources of meteorological data. Further information on the use of specific types of meteorological data in NAME, as well as more detailed discussions of these data sets, are supplied in additional documentation referenced from within this section.

⁷The phrase ‘met module’ is often used for brevity when, strictly speaking, the object being referred to is really an ‘instance of a met module’ (and similarly with the difference between a ‘flow module’ and an ‘instance of a flow module’). To clarify the distinction here, a ‘met module’ refers to a generic class of data structures, properties, methods, etc. for use with a particular category of meteorological data, whereas an ‘instance of a met module’ refers to a particular instance of a met module within such a class. The discrimination is a somewhat academic one, and in practice NAME users tend to refer to ‘met modules’ and ‘flow modules’ without confusion.

1.6.1 Using observations from a weather station (single-site meteorology)

NAME can be run with a local description of the meteorology and atmospheric structure supplied by a *single-site meteorology*. The single site met data might consist of actual observations recorded at a weather station, pseudo-observations (e.g. extracted from an NWP modelling system), or user-supplied estimates of local weather parameters, and could include forecast meteorology as well as current or past meteorological conditions. Alternatively, rather than using the actual meteorology (or estimates of the actual meteorology), the user can equally decide to consider idealised meteorological scenarios here (for instance, as an approach to studying the sensitivities of dispersion predictions to various meteorological parameters).

The single-site meteorology capability assumes horizontal homogeneity of the met and flow fields and is therefore intended only for short-range dispersion problems (that is, dispersion on spatial scales where horizontal changes and variations in the meteorology are expected to be small enough to ignore). In practice, this length scale is likely to be of the order of a few tens of kilometres, say, in regions where topographical effects are small (and where changes in the meteorology are due to mesoscale and synoptic scale influences), but can be much shorter in areas of complex terrain or where there are abrupt changes in the terrain such as along coasts.

The framework for single-site meteorology within NAME is similar to the scheme used in the ADMS dispersion model from Cambridge Environmental Research Consultants, CERC (see, e.g., Thomson [2012]). The single-site met ‘pre-processor’ that constructs met and flow parameters from the available input data is essentially the same in both models and, with a few exceptions, input met data files are also the same. These input files comprise of a header section describing the meteorological parameters that are present in the data file, followed by a series of data records defining the evolution of these parameters at an hourly frequency. As with ADMS met input files, missing data values are allowed. There are, however, some differences between NAME and ADMS concerning their treatment of single-site meteorology. The most fundamental difference is that NAME treats each data record as a representation of the instantaneous meteorological conditions at the start of the hour and will interpolate (in time) over the course of that hour using the next data record in the sequence. Conversely, ADMS considers each hour in the series in a discrete sense and computes an individual dispersion realisation for each data record separately. One consequence of this difference in approach is that plumes are ‘straight’ in ADMS (because of the fixed meteorology for each separate hour), whereas they are generally ‘curved’ in NAME (as the plume responds to gradual changes in the meteorology over the course of each hour). A further difference with ADMS is that NAME does not include a frequency-based approach for assessing dispersion climatologies, in which a defined set of scenarios can be modelled and results then merged using a corresponding set of frequencies (although, in principle, such an approach would be possible with NAME by post-processing output in a suitable way). One other difference between NAME and ADMS is that NAME supports a time-zone option for its single-site met data files allowing the reference time frame

to be specified for each data record (e.g. 'UTC' or 'UTC+01:00'). When not specified, the time is assumed to be the local solar time (as in ADMS).

The single-site met module in NAME is designed to read single-site met data files and store the information. The single-site flow module is then designed to use the available single site met data to construct mean flow profiles and add turbulence information. In principle, multiple instances of the single-site met module and single-site flow module can be defined (with each instance of the single-site flow module using input from a single instance of the single-site met module). In practice, however, only one instance of these met and flow modules is usually defined in any particular NAME run. Note that if multiple instances were to be defined (e.g., to cover different spatial domains around neighbouring weather stations), the flow information is not blended between these domains and there can be an abrupt change in flow characteristics across the boundary between them. Setting up NAME to use single-site met data in this way is not recommended (however there are some situations where multiple met and flow module instances are defined and used in a self-consistent manner). There is also a similar issue when nesting a single-site flow module within an NWP flow module, and although this modelling configuration is possible within NAME it is again generally not recommended.

Single-site meteorology assumes a horizontally homogeneous flow with a flat topography (topographic height is taken to be at sea level, and the surface pressure to be the sea level pressure in the ICAO standard atmosphere). Any terrain and coastal effects are ignored.

Instances of the single-site met and flow modules are defined in practice using the two blocks, **Single Site Met Module Instances** and **Single Site Flow Module Instances**, in the main input file. The **Single Site Met Module Instances** block defines the characteristics for each set of single-site met data to be read (as stated above, there is usually only one instance defined in a NAME run, although it is possible for more than one set to be read here).

Each set of met data has an associated **Name** (the name of the met module instance). The location of the meteorological site should be specified by a longitude **Long** and latitude **Lat** in a longitude-latitude based coordinate system **H-Coord** (which usually refers to the standard longitude-latitude coordinates of the site in degrees). The standard height of a wind measurement at the meteorological site is 10 m above ground level (m a.g.l.) but a non-standard height can be used by modifying the value of the input variable **Height**. The roughness length at the met site **z0** and in the dispersion area of interest **z0D** are required (and can be the same), together with an indication as to whether the value at the met site is representative of the area for which dispersion calculations are required. If the logical keyword **Representative?** is set to **Yes** then the values for **z0** and **z0D** should be the same and no distinction is made between the met site and dispersion site. On the other hand, if **Representative?** is set to **No** then a correction is applied to the vertical profiles to account for the different roughness length to be used in the dispersion calculations compared with the value appropriate to the input met data. The name of the input file containing hourly met data is supplied in the variable **Met File** (including the folder path where needed). It is also possible for the

user to specify their own values to use in the unresolved mesoscale motions and free tropospheric turbulence parametrizations, rather than the default values, see [TURBULENCE SCHEMES USED IN NAME](#) in 1.7.4.3 for further details. Finally, the logical flag [Ignore Fixed Met Time?](#) gives the user the option to ignore the actual Fixed Met Time in a Fixed Met run and to use the first met record in the met file instead, see [USING 'FIXED' METEOROLOGY IN NAME](#) in 1.2.7.4.

Once instances of the single-site met module have been set up, then instances of the single-site flow module can be defined to use these single-site met data. The [Single Site Flow Module Instances](#) block defines the characteristics of the single-site flow data (and includes details of the single-site met data to be associated with each flow instance, and their domain of validity). Once again, there is usually only one instance of a single-site flow module defined in any NAME run, although it is possible for more than flow instance to be declared.

Each instance of the single-site flow module has a [Name](#) and is associated with input data from a single instance of the single-site met module, which is specified by setting the keyword [Met Module](#) as [Single Site Met](#) and the keyword [Met](#) to the name of the single-site met module instance. The domain of validity of the flow module instance is established by giving the name of the domain in the keyword [Domain](#).

1.6.2 Using meteorological fields from a Numerical Weather Prediction model

Using single-site meteorology in NAME has its limitations as it does not represent horizontal variations in the flow field and might not always capture the correct vertical structure of the atmosphere, especially in complex meteorological situations (e.g. the Buncefield oil depot fire). For dispersion problems beyond the local scale (and even for local-scale problems too), a more detailed treatment of the meteorology is often desirable. This can be achieved by using gridded meteorological fields from Numerical Weather Prediction (NWP) models, referred to as *NWP meteorology*.

NAME can be configured to use output fields from various NWP models (e.g. the Met Office Unified Model, MetUM, or the ECMWF Integrated Forecast System, IFS) and for different configurations of these models (e.g. Global and UK 1.5 km configurations of the MetUM). These NWP data sets can be historical reanalyses, analysed meteorology⁸ and forecast meteorology for prediction into the future. Spatial resolution is an important factor when considering the utility of an NWP data set, as it determines the scales that can be represented within the model simulation (e.g. global scale models are unable to resolve the fine-scale details in areas of complex terrain). A full description of the use of NWP data sets in NAME is supplied in the separate documentation [» MD15/7: USING NWP DATA IN NAME - A PRACTICAL GUIDE «](#).

NAME uses a (one-way) offline coupling with the NWP model. Here the basic principle is that the gridded fields required by NAME are written out from the NWP model at a regular frequency as

⁸'Analysis' met data sets for NAME typically consist of a sequence of short-range forecasts from successive cycles of the forecast model.

binary data files. These files are then read by NAME to recreate an approximate⁹ representation of the evolution of the NWP model simulation for use in the dispersion modelling. Topography (orography) and other ancillary fields from the NWP model might also be used in NAME (topography is a compulsory field when using NWP meteorology, whereas other ancillaries are only required when specific functionality is requested; see 1.6.2.1 below).

The contents and structure of a set of NWP data files are described by an associated *NWP met definition file*, which can be thought of as supplying the meta-data that characterises the NWP met data. This ‘meta-data’ includes, for instance, a description of the NWP parameters that are present in the met files and their corresponding horizontal grids and vertical levels. For most users, NWP met definition files can usually be regarded as a ‘black-box’ and it is sufficient to simply reference the met definition file (or files) appropriate for the set (or sets) of NWP data being used in their NAME run. Advanced users intending to utilise their own NWP met data in NAME will need to understand how to set up an NWP met definition file, see [▶ MD15/7 ◀](#) for further guidance on this.

The NWP met module in NAME is designed to read NWP met data files and store the information. The NWP flow module is then designed to use this NWP met data to construct flow information (and cloud, rain, etc.) as and when requested by other model components within NAME. Multiple instances of the NWP met module and NWP flow module can be defined (with each instance of the NWP flow module taking input data from a single instance of the NWP met module). Multiple instances of the NWP met and flow modules are often associated with the nesting of NWP data sets (that is, the preferential use of high-resolution meteorological fields where these are available within a sub-region of low-resolution fields) and also the use of domain-decomposed meteorological fields (for example, the global ‘cut-out’ domains for the UM).

Instances of the NWP met modules and NWP flow modules are defined using the two blocks **NWP Met Module Instances** and **NWP Flow Module Instances**, respectively, in the main input file. The **NWP Met Module Instances** block gives instructions to NAME for each set of NWP met data to be read, see Table 1.18 for details. Each NWP met module instance defined in a NAME simulation is identified by a unique user-specified name given by the input keyword **Name** which, although arbitrary, should be chosen to be descriptive of the NWP met data being represented. On the other hand, the keyword **Met Definition Name** refers to the name of the NWP met definition and needs to match the name specified in the corresponding met definition file.

Various controls can be implemented in terms of how the NWP met data files are read and how certain meteorological parameters are subsequently treated in NAME (specifically the boundary layer depth). For instance, the user can select whether to use the values of boundary layer depth given in the NWP data directly (which is the default setting and is usually recommended for most types of NWP data) or whether to recalculate the boundary layer depth within NAME based on

⁹For practical reasons, NAME met files only contain a subset of the full collection of NWP model variables and these are output at a slower frequency than the internal NWP model time step. In principle, if every variable required by NAME were to be output at the native model time step then an exact recreation of the NWP model state would be possible inside NAME, but this would be prohibitively expensive in practice!

Variable name	Description
Name	Name of met module instance (user-specified)
Met Definition Name	Name of NWP met definition (as specified in its met definition file)
Use NWP BL Depth?	Indicates that the NWP values of boundary layer depth are to be used
Min B L Depth	Minimum and maximum limits to be imposed on boundary layer depth
Max B L Depth	
Restore Met Script	User script for restoring working copy of met data files from an archive copy
Delete Met?	Indicates that working copy of met data files will be deleted after use
Topography Folder	Folder containing topography data file
Met Folder	Folder containing met data files (for non-ensemble meteorology)
Met Folders	Array of folders containing ensemble met data (one ensemble member per folder)
Met Folder Stem	Stem of folders containing ensemble met data (one ensemble member per folder)
Ensemble Met Folder	Folder containing ensemble met data (all ensemble members contained in same met file)
Mesoscale SigU	Horizontal velocity standard deviation for use in the parametrization of unresolved mesoscale motions
Mesoscale TauU	Horizontal Lagrangian timescale for use in the parametrization of unresolved mesoscale motions
Free Trop SigU	Horizontal and vertical velocity standard deviations in free troposphere
Free Trop SigW	
Free Trop TauU	Horizontal and vertical Lagrangian timescales in free troposphere
Free Trop TauW	
Update On Demand?	Indicates that met module instance is to be updated using 'update-on-demand'

Table 1.18: Keyword variables defining an NWP met module instance

other input fields describing the thermodynamic profile of the atmosphere (which may be more appropriate for older UM data sets). This choice is governed using the logical switch [Use NWP BL Depth?](#) (default is [Yes](#)). It is also possible to apply minimum and maximum limits on the boundary layer depth within NAME using the input variables [Min B L Depth](#) and [Max B L Depth](#).

NAME will attempt to read NWP met data files and topography files from the specified folders [Met Folder](#) and [Topography Folder](#), respectively (with the exception of met files from ensemble NWP systems which are treated somewhat differently, see [USING ENSEMBLE NWP METEOROLOGICAL DATA](#) in 1.6.2.2 for further details). NWP met data for NAME are typically stored in a separate archive folder distinct from the working met folder specified by [Met Folder](#). This is especially true when using historical met data sets, which usually need to be stored in gzipped form due to data volume constraints, but which need to be unzipped prior to reading by NAME. The recommended practice here is to copy each met file from the archive to the working met folder on request by NAME and to unzip this local copy. Then, once NAME has finished reading this particular met file, the local copy of the file can be deleted. This approach keeps the original archive files unchanged and helps to prevent corruption of the data. The restore and subsequent deletion of NWP met data files can be automated using the two input variables [Restore Met Script](#) and [Delete Met?](#). The first variable [Restore Met Script](#) enables the user to supply a script for restoring a local copy of a met file from an archive on demand. Such a script is generally referred to in NAME as a *met restore script* and it can be written flexibly according to the requirements (folder structures, etc.) of a user. The second variable [Delete Met?](#) is a logical flag indicating whether NWP met files in the working met folder are to be deleted after use.



Users should take extra care when invoking the [Delete Met?](#) option as, when set to [Yes](#), NAME will attempt to delete met files from the working met folder once it has read them. Therefore ensure that the working met folder is always set to a local temporary folder space and is not pointing at the main archive folder. Deletion of met files by NAME might be prevented by setting read-only access permissions on the files and folder (which may offer some protection against accidental deletion of the archive) but this safeguard should not be relied upon.

NAME has the ability to only read an input met file and process the met and flow information when that information is actually required by some other component of the model and, in particular, met files will not be read if there is no requirement to do so. Such functionality is referred to as 'update-on-demand' and can significantly reduce the computational overhead associated with the reading and processing of met data. The update-on-demand capability is especially useful with some of the modern NWP datasets from the Unified Model that are domain-decomposed into multiple subregions (or ParTs), as only those parts that are needed by NAME will be loaded. The user is able to request the update-on-demand functionality by setting the logical keyword [Update On De-](#)

[mand?](#) to [Yes](#) in both the NWP met module instance and NWP flow module instance. If these are set to [No](#) (which is the default if not specified), then the met and flow modules will be processed in full at each met update step (irrespective of whether that information is later used or not).



Note that it is not possible to use met-on-demand functionality together with a parallel dedicated I/O thread for reading NWP met data files from disk and processing the NWP fields ahead of schedule.

The user can control various aspects of the turbulence and unresolved mesoscale motions parametrizations used in NAME via keywords in the [NWP Met Module Instances](#) input block. The variables [Mesoscale SigU](#) and [Mesoscale TauU](#) allow the user to define their own values for the horizontal velocity standard deviation and horizontal Lagrangian timescale to be used in the parametrization of unresolved mesoscale motions. The default values are dependent on the resolution of the input NWP met data and are specified in the [NWP Met Definitions](#) block in the met definition file. The default values (see Table 1.19) are assumed when no user values are specified. The variables [Free Trop SigU](#), [Free Trop SigW](#), [Free Trop TauU](#) and [Free Trop TauW](#) can be defined for the horizontal and vertical velocity standard deviations and Lagrangian timescales in the free troposphere, for use in the free-tropospheric turbulence parametrization. Default values of 0.25 m s^{-1} , 0.1 m s^{-1} , 300 s and 100 s, respectively, are assumed if no values are specified by the user. Further details on the parametrizations of turbulence and unresolved mesoscale motions can be found in [TURBULENCE SCHEMES USED IN NAME](#) in section 1.7.4.3.

Model / Resolution	Standard deviation (m s^{-1})	Lagrangian timescale (s)
Global / $> 60 \text{ km}$, 3-hourly	0.95	10000
Global / $\sim 40 \text{ km}$, 3-hourly	0.9	10000
Global / $\sim 20 \text{ km}$, 3-hourly	0.8	10000
Mesoscale / $\sim 10 \text{ km}$, 3-hourly	0.8	10000
Mesoscale / $\sim 10 \text{ km}$, hourly	0.7	8000
4km or UKV / $\leq 4 \text{ km}$, hourly	0.55	6500

Table 1.19: Default parameters for the parametrization of unresolved mesoscale motions

Once instances of the NWP met module have been set up, then instances of the NWP flow module can be defined to use these NWP met data. The [NWP Flow Module Instances](#) block defines the characteristics of the NWP flow data (and includes details of the NWP met data to be associated with each flow instance, and their domain of validity), see Table 1.20 for details.

Each instance of the NWP flow module is associated with input NWP data in a single instance of the NWP met module by setting the variable [Met Module](#) as [NWP Met](#) and [Met](#) to the name of the NWP met module instance. The domain of validity of the flow module instance is set by giving the name of the domain (as specified in its met definition file) in the keyword [Domain](#). The variable [Ancillary Met Module Instances](#) allows a list of ancillary met modules, as defined within

Variable name	Description
Name	Name of flow module instance (user-specified)
Met Module	Name of associated met module
Met	Name of associated met module instance
Domain	Name of the domain of the flow module instance
Ancillary Met Module Instances	Name(s) of associated ancillary met module instances
Update On Demand?	Indicates that flow module instance is to be updated using 'update-on-demand'

Table 1.20: Keyword variables defining an NWP flow module instance

an **Ancillary Met Module Instances** block, to be (optionally) included within the NWP flow module instance (although ancillary met modules are often not added to the flow module). Further details on using ancillaries within NAME can be found below in [USING LAND SURFACE FIELDS FROM NWP ANCILLARIES](#) in 1.6.2.1.

1.6.2.1 Using land surface fields from NWP ancillaries

Ancillary fields from an NWP model can be read into NAME to provide additional input data that varies on a slower timescale than the standard met fields contained in the input meteorological files. Ancillary data is therefore normally assumed to be fixed or to vary on a monthly timescale. The model topography (also referred to as the orography) is a special type of ancillary field which is treated differently in NAME to other ancillary fields described here. In general, ancillary fields provide surface information that is used by NAME for specific types of model runs (dust runs, runs using the land-use dependent dry deposition scheme and runs using the urban scheme).

The ancillary fields are split across three **Flow Attributes**:

Plant contains ancillaries on vegetation characteristics required for the land-use dependent dry deposition scheme,

Soil contains ancillaries on soil characteristics required for the dust scheme, and

Surface contains ancillaries on other surface characteristics required for dust runs and runs using the urban scheme or the land-use dependent dry deposition scheme.

A NAME run can include none, some or all of the Plant, Soil and Surface flow attributes. Table 1.21 lists the ancillary field names recognised in NAME and the attribute to which each belongs. Note that for each attribute all of the information for that attribute must be provided. Soil moisture and canopy height are, however, optional ancillaries as these can be supplied by the NWP meteorological input instead.

NAME is currently designed to use ancillary information from the Met Office Unified Model (with nine specific land use types from the MOSES land surface scheme, and six specific dust size bins). Users wishing to use alternative ancillaries are recommended to first seek advice and guidance.

Ancillary Field Name	Flow Attribute
land fraction	Surface
land use fractions (9 land use types)	Surface
soil moisture in layer (kg/m ²)	Surface
clay mass fraction	Soil
silt mass fraction	Soil
sand mass fraction	Soil
soil particle mass fractions (6 size bins)	Soil
leaf area index (5 plant land use types)	Plant
canopy height (m) (5 plant types)	Plant

Table 1.21: NAME ancillary fields

To use ancillary data with NAME, the appropriate flow attribute(s) should be included in the **Flow Attributes** block of the main input file. This is done by adding suitable entries with the keyword **Name** set to *Surface*, *Soil* or *Plant* as required. The ancillary information is included in an NWP flow module instance by specifying the names of the relevant **Ancillary Met Module Instances** in the **NWP Flow Module Instances** block. Multiple ancillary met module instances can be given as a semi-colon separated list of names, each corresponding to an ancillary file (or set of monthly ancillary files) containing one or more ancillary fields. Each such ancillary met module must be defined and correspond to an ancillary met definition and an ancillary met file structure definition. These are similar to their NWP equivalents usually specified in the met definition file.

Ancillary met module instances are defined in the **Ancillary Met Module Instances** input block. The name of the ancillary met module (as referenced by the **NWP Flow Module Instances** block) is specified using the keyword **Name**, the folder containing the ancillary data files is given using the variable **Met Folder** and the name of the corresponding ancillary met definition is specified using the variable **Met Definition Name**. The ancillary met definitions are defined in the **Ancillary Met Definitions** block, which should specify the following variables.

Name	gives the name of the ancillary met definition
Binary Format	sets the binary format of the ancillary data files (this is usually <i>BIG_ENDIAN</i>)
File Type	sets the file type of the ancillary data files (this is usually <i>PP</i>)
Dt	specifies the time interval between ancillary fields, which can be <i>infinity</i> for fixed ancillaries or <i>monthly</i> for ancillaries varying on a monthly timescale
Prefix	sets the prefix for the filename(s) of the ancillary file(s)
Suffix	sets the suffix for the filename(s) of the ancillary file(s) (this is usually <i>pp</i>)
Met File Structure Definition	gives the name of the corresponding ancillary met file structure definition block to use
H-Grid	specifies the name of the horizontal grid for the ancillary fields

The horizontal grid for ancillary fields can either be the same as, or different from, a horizontal grid used by the NWP meteorological fields. Users should ensure, however, that the horizontal grid for the ancillary data is appropriately defined when it is different from an NWP grid (and is therefore not defined as part of the NWP definition file). Ancillaries with the same flow attribute (that is, plant, soil or surface) must use the same horizontal grid, but in principle it should be possible for the different flow attributes to use different grids (e.g. the H-Grid for the plant ancillaries, say, might be different to that for the surface ancillaries), although this aspect of ancillaries would need further testing (as the UM ancillaries that are used are usually on the same horizontal grid throughout). For fixed ancillaries (where **Dt** is set to *infinity*), the (single) ancillary file has the name *Prefix.Suffix*. For monthly ancillaries (where **Dt** is set to *monthly*), the (multiple) ancillary files are named *Prefix{mm}.Suffix*, where {mm} is the two-digit month number (that is, '01' for January through to '12' for December).

The **Ancillary Met File Structure Definition** is a named block describing the structure of an ancillary file. The column keyword **Field Name** gives the name(s) of the ancillary field(s) stored in a file. Allowed ancillary field names are listed in Table 1.21. The variables **Lowest Level** and **Highest Level** specify the lowest and highest model 'levels', respectively, of each ancillary field. Despite the fact that ancillary fields are all surface fields, some are stored in NAME on pseudo vertical levels (e.g., the nine land-use types are stored as nine pseudo levels). The variable **Field Code** is the field (or UM stash) code and **3-d?** denotes whether the field is part of a three-dimensional NWP / ancillary field (this is usually set to *No* for an ancillary field).

Additional documentation on the use of UM ancillaries in NAME will be prepared for future release. In the meantime, please contact us for further information on the appropriate values to give for variables in the **Ancillary Met File Structure Definition** input block when using the UM ancillaries.

1.6.2.2 Using ensemble NWP meteorological data

Meteorological uncertainties have long been acknowledged in the field of weather forecasting, especially when assessing risks from high-impact severe weather events. Over recent decades NWP ensemble systems have become an important tool in operational forecasting for quantifying such uncertainties. They provide guidance on the general level of confidence in the forecast evolution (e.g. in support of deterministic products), help to identify possible alternative outcomes (including high-impact weather events) and facilitate a probabilistic approach to weather prediction. They were initially targeted at medium-range forecasting on global scales (e.g. the operational ensemble prediction systems at ECMWF and NCEP), but in recent years the approach has been applied to shorter-range ensemble prediction at regional and local scales (e.g. the Met Office MOGREPS system, where a 2.2 km resolution ensemble has recently been introduced over the UK area).

For dispersion modelling, a quantitative assessment of uncertainties becomes a far more difficult challenge. This is, in part, because other sources of uncertainty are also present in addition to those associated with the input meteorology. For instance, there are uncertainties in the specification of the initial distribution of a release and also in the model representation of various processes involved in the transport and dispersion. Uncertainties in dispersion predictions have traditionally been judged in a qualitative way, with, for example, an assessment of the influence of meteorological uncertainties on the dispersion outcome being based on the confidence in the weather forecast and identification of possible sources of error in the forecast evolution. However, with the availability of ensemble NWP data, NAME has a capability to generate multiple realisations of the dispersion with each realisation based on a single member of the ensemble forecast. Statistical processing within NAME can then yield products such as ensemble mean, the median over the ensemble (and other percentile measures), and the level of agreement on exceeding thresholds.

Utilising NWP ensemble forecasts directly within NAME aims to assess the impact on dispersion of the meteorological uncertainty conveyed in the NWP ensemble (while accepting that other sources of uncertainty in the dispersion problem will not be represented). In other words, it enables an assessment to be given for that part of the uncertainty in the dispersion predictions that propagates from uncertainties in the meteorological evolution at the scales resolved by the NWP ensemble system (while acknowledging that these might not 'see' local flow features). The degree of confidence in the NAME predictions and possible outlier solutions can be identified. In practice, NAME has been used with ECMWF EPS forecasts to look at uncertainty in long-range transport, and with an earlier version of MOGREPS for regional-scale dispersion problems.



An EPS-based NAME run can supplement a deterministic 'best-estimate' prediction with information quantifying the uncertainty arising from the meteorology.

Running NAME using an ensemble of weather realisations presents new challenges, both of

a technical nature and of interpretation of model results. Firstly, an EPS-based NAME run is essentially a set of multiple realisations of the dispersion arising from using the individual members of the weather prediction ensemble in sequence. As a consequence, dispersion calculations need to be repeated for every member of the ensemble (or subset of an ensemble) in turn, which has a consequent computational burden. To a first approximation, an n -member ensemble will require an n -fold increase in computational time over that of a single realisation. The current design of the NAME code implements the ensemble 'loop' at an outermost level (the loop over 'cases') and all statistical outputs calculated over the ensemble are constructed and output as a final processing step. This sequential approach (running on a single computer) implies that an n -member ensemble will require approximately n times the clock time for that of a single realisation. An alternative strategy here might be to use a scheduler script to distribute multiple NAME runs (possibly across multiple servers) to model the individual realisations and then combine the resulting output fields in an off-line processing step. This latter technique would offer the benefit of greater freedom when allocating available computer resources to a problem.

The interpretation of NAME ensemble results also has some subtleties, which means results need to be communicated carefully (although these are similar to some of the issues that arise from use of EPS forecasts in weather-related applications and similar fields, and much can be learnt from existing uses of EPS forecasts in these other disciplines). Statistical quantities such as percentiles (calculated over the ensemble sample) and probabilities of exceeding various threshold values can be useful for distilling the large amount of information contained within the full set of ensemble predictions into a more usable form.

Percentiles are helpful for viewing the range of possible outcomes at individual locations (e.g. in a 'meteogram'-style time series product) but can be potentially misleading if interpreted as a deterministic-type prediction (as is often done with, e.g., the ensemble median). The same argument applies to the ensemble-mean prediction, which is likely to exhibit good performance in a statistical sense (at least when applying a root-mean-square-based metric) but does not provide a physically realistic dispersion solution. Care must be taken not to interpret statistical fields too literally as 'real' solutions. In the extreme case here, the 100-th sample percentile (maximum value over the ensemble) can be useful as a measure of the 'worse case' at any specific location, but such an outcome will not occur simultaneously across the entire domain. The danger of presenting a 100-th percentile concentration map is in its potential misinterpretation that this scenario could actually occur universally. However, conversely, the complement of the 100-th percentile map (that is, the region in which there is a high level of confidence that material is below specified thresholds) could be quite useful as an indication of 'safe' areas (e.g. for evacuation).

Arguably the best use of ensemble prediction is to describe event probabilities (e.g. the likelihood of exceeding a specified concentration threshold). Again some care is needed here to ensure a correct interpretation of precisely what is meant by an event 'probability' in this context. A more accurate description would be the probability, or likelihood, *conditional on* the assumptions underly-

ing the modelling, which consider only the component of uncertainty in the meteorological evolution as conveyed in the EPS forecast. Since no efforts are being made to treat other sources of uncertainty in the dispersion prediction (especially in the source description) then these will not, in general, be the same as the ‘true’ (or calibrated) probabilities of the event occurring. In other words, the probabilities are not expected to be well-calibrated (as they might be expected to be for, say, a temperature or rainfall forecast). This is not to say that the calculated probabilities are not useful, because they do accurately portray uncertainty within one aspect of the dispersion problem (i.e. the confidence in the meteorological forecast). However in communicating results, one needs to ask whether it is sensible to talk about quantitative probabilities when some of the underlying uncertainties in the modelling may be poorly understood, and perhaps it is better to use terms such as ‘level of confidence’ or ‘level of agreement’ instead? For a fuller discussion on the use of ensemble NWP forecasts in NAME and their interpretation, see Jones [2011].

An ensemble of NWP forecast realisations are represented as multiple ‘cases’ in a NAME run by specifying the required number of cases (i.e., ensemble members) in the keyword variable [Met Ensemble Size](#) in the [Multiple Case Options](#) block of the main input file. Normally these ensemble members would be the individual forecast members produced by an ensemble prediction system, although they do not necessarily have to be created in this way. It is equally possible to use a set of time-lagged forecasts from a (deterministic) forecasting system, or even a set of forecasts from a collection of different (deterministic) forecasting systems. The latter two scenarios are examples of what are referred to as a *poor-man’s ensemble*. NAME supports a flexible approach towards the organisation and structure of ensemble met data files: different ensemble members can be distributed as files in different met folders, or alternatively all ensemble members can be stored within a single set of met files in one folder. The appropriate behaviour in NAME is controlled by setting *precisely one* of the ensemble met folder options as below.

Met Folders	defines an array of folder names (with one ensemble member per folder)
Met Folder Stem	defines a folder stem for a collection of folders (the actual folders are then suffixed as ‘1’, ‘2’, etc., with one ensemble member per folder)
Ensemble Met Folder	defines a single folder containing ensemble met data (with all ensemble members contained in the same met file)

Output fields can be requested for each individual ensemble member (and are identified by the case identifier) or they can be statistical quantities (ensemble mean, percentiles or probabilities) calculated over the ensemble of cases. See [STATISTICAL PROCESSING](#) in 1.4.1.3 for further details on requesting statistical outputs from an ensemble of cases.

1.6.3 Using rainfall estimates from weather radar

An accurate representation of rainfall patterns at high spatial and temporal resolution is important when modelling wet deposition, especially for identifying deposition ‘hot-spots’ where a plume has been intercepted by localised intense rainfall.

NAME has an option to use rainfall estimates obtained from a weather radar network or from a rainfall analysis system assimilating rainfall observations with modelled fields. When invoked, rainfall information is read from a collection of input data files (as with NWP met files, any radar met files are labelled by their time stamp) and these rainfall estimates are used in preference to any NWP precipitation fields that might also be available.

The use of radar or processed rainfall analysis data in NAME follows the same general principles as when using NWP met data, and much of the functionality described in this section is similar to that discussed above in [USING METEOROLOGICAL FIELDS FROM A NUMERICAL WEATHER PREDICTION MODEL](#) in 1.6.2. For example, all structural information about a set of radar met data is stored as part of a radar met definition, and also when using the radar met option there is the ability to restore radar met files from an archive and then delete the local copy after use.

The contents and structure of a set of radar met data files are described by an associated *radar met definition file*, which can be thought of as supplying the ‘meta-data’ that characterises the radar met data. This meta-data includes, for instance, a description of the parameters that are present, their horizontal grid and the temporal frequency of the data. For most users, radar met definition files can usually be regarded as a ‘black-box’ and it is sufficient to simply reference the met definition file (or files) appropriate for the set (or sets) of radar met data being used in their NAME run.



Currently a radar met definition only expects a single field to be present in a radar met file, which should be either an instantaneous precipitation rate (in units of mm/hr) or a one-hour precipitation accumulation (in mm). Although NAME would, in principle, be able to read other types of field here (such as precipitation type, cloud cover, snow probability, etc.), there is presently no code in place within NAME to utilise such additional fields.

The radar met module in NAME is designed to read radar met data files and store the information. The radar flow module is then designed to use this radar data to construct rain information as and when requested by other model components within NAME. Note that the ‘Radar Met’ and ‘Radar Flow’ module types currently only provide flow information with the ‘Rain’ attribute (although in the future this might extend to other flow and cloud information too). Multiple instances of the radar met module and radar flow module can be defined, if needed, with each instance of the radar flow module taking input data from a single instance of a radar met module. The [Flow Order](#) blocks in the main input file determine the precedence in which any multiple instances of a radar flow module are applied in a NAME run.

Instances of the radar met modules and radar flow modules are defined using the two blocks **Radar Met Module Instances** and **Radar Flow Module Instances**, respectively, in the main input file. The **Radar Met Module Instances** block gives instructions to NAME for each set of radar met data files to be read, see Table 1.22 for details. Each radar met module instance defined in a NAME simulation is identified by a unique user-specified name given by the input keyword **Name** which, although arbitrary, should be chosen to be descriptive of the radar met data being represented. On the other hand, the keyword **Met Definition Name** refers to the name of the radar met definition and needs to match the name specified in the corresponding met definition file.

Variable name	Description
Name	Name of radar met module instance (user-specified)
Met Definition Name	Name of radar met definition (as specified in its met definition file)
Restore Met Script	User script for restoring working copy of radar data files from an archive copy
Delete Met?	Indicates that working copy of radar data files will be deleted after use
Met Folder	Folder containing (working copy of) radar data files
Update On Demand?	Indicates that met module instance is to be updated using 'update-on-demand'

Table 1.22: Keyword variables defining a radar met module instance

Various controls can be implemented in terms of how the radar met data files are to be read. NAME will attempt to read radar met data files from the specified folder **Met Folder**. Radar met data for NAME are typically stored in a separate archive folder distinct from the working met folder (this is especially the case when using historical radar data sets). The recommended practice here is to copy each radar met file from the archive to the working met folder on request by NAME. Then, once NAME has finished reading this particular radar met file, the local copy of the file can be deleted. This approach keeps the original archive files unchanged and helps to prevent corruption of the data. The restore and subsequent deletion of radar met data files can be automated using the two input variables **Restore Met Script** and **Delete Met?**. The first variable **Restore Met Script** enables the user to supply a script for restoring a local copy of a radar met file from an archive on demand. Such a script is generally referred to in NAME as a *met restore script* and it can be written flexibly according to the requirements (folder structures, etc.) of a user. The second variable **Delete Met?** is a logical flag indicating whether radar met files in the working met folder are to be deleted after use.



Users should take extra care when invoking the [Delete Met?](#) option as, when set to [Yes](#), NAME will attempt to delete the radar met files from the working met folder once it has read them. Therefore ensure that the working met folder is always set to a local temporary folder space and is not pointing at the main archive folder. Deletion of met files by NAME might be prevented by setting read-only access permissions on the files and folder (which may offer some protection against accidental deletion of the archive) but this safeguard should not be relied upon.

The user is also able to request 'update-on-demand' functionality by setting the logical keyword [Update On Demand?](#) to [Yes](#) in both the radar met module instance and radar flow module instance. This is similar to the other met modules and means that NAME will only read an input radar met file and process the met and flow information when that information is actually required by some other component of the model and, in particular, radar met files will not be read if there is no requirement to do so. If these keywords are set to [No](#) (which is the default if not specified), then the met and flow modules will be processed in full at each met update step (irrespective of whether that information is later used or not).

Once instances of the radar met module have been set up, then instances of the radar flow module can be defined to use the radar met data. The [Radar Flow Module Instances](#) block defines the characteristics of the radar flow (and includes details of the radar met data to be associated with each flow instance, and their domain of validity), see Table 1.23 for details.

Variable name	Description
Name	Name of radar flow module instance (user-specified)
Met Module	Name of associated met module (which is Radar Met)
Met	Name of associated met module instance
Domain	Name of the domain of the flow module instance
Update On Demand?	Indicates that flow module instance is to be updated using 'update-on-demand'

Table 1.23: Keyword variables defining a radar flow module instance

Each instance of the radar flow module is associated with the radar data in a single instance of the radar met module by setting the variable [Met Module](#) as [Radar Met](#) and [Met](#) to the name of the radar met module instance. The domain of validity of the flow module instance is set by giving the name of the domain (as specified in its radar met definition file) in the keyword [Domain](#).

1.6.3.1 Using precipitation fields from the UKPP and EuroPP gridded analysis systems

The specific radar definition files supplied with NAME enable it to read radar-derived precipitation analysis fields from the Met Office's **UKPP** and **EuroPP** nowcasting systems (Wright [1994]). The UKPP system provides 2 km gridded fields over the UK area, whereas the EuroPP gives 5 km fields over a wider European domain.

The radar met data files contain instantaneous precipitation rate or one-hour precipitation accumulations (here the one-hour accumulation in mm can be considered as a mean rate over the past one-hour period because the precipitation field expected by NAME is in units of mm/hr). Use of accumulations (time averages) is advised for high spatial resolution data sets when the temporal resolution is limited and precipitation features have a tendency to advect over several grid lengths between each successive pair of instantaneous time frames. Use of time-averaged precipitation in these circumstances can overcome this 'rain-hopping' problem.



When using radar-derived precipitation fields, it is advisable to use radar met fields at a similar spatial resolution to the NWP fields being used. For instance, the UKPP (2 km) data set would be best suited when running NAME with the UKV (1.5 km) configuration of the Unified Model.

1.6.4 Advanced options for flow information

Some advanced flow modelling options are available within NAME and are described below. These are generally intended for research use or are presently in a development phase and might not be fully functional. Users should seek further advice before attempting to use any of these additional modelling features.

1.6.4.1 Using the linear flow model LINCOM with NAME

LINCOM is a linearised flow model developed by the Risø National Laboratory in Denmark for representing flows over complex terrain. The model calculates perturbations to a background mean flow caused by variations in ground elevation and local surface roughness within its domain. The version of LINCOM that has been interfaced with NAME is restricted to flows with neutral stability.



The LINCOM flow module within NAME is designed only to provide an *interface* to the LINCOM model. The LINCOM model itself is third-party software owned by the Risø National Laboratory and is not supplied with the NAME distribution. Any user interested in exploiting the LINCOM functionality within NAME should enquire directly with Risø regarding licensing and supply of LINCOM software.

Control of LINCOM from within NAME is provided through a **LINCOM Flow Module Instances** block in the set of main input files. The user needs to supply suitable files of terrain data and/or

roughness length data in order to calculate flow perturbations caused by terrain and roughness, respectively. When used, a stand-alone LINCOM executable is invoked at each meteorological update step to perform off-line flow calculations, with the flow data then subsequently read back into NAME. The background (unperturbed) meteorology used for the LINCOM calculations is obtained from a suitable large-scale flow module (single-site flow or NWP flow) in NAME.

1.6.4.2 Modelling the flow around an isolated building

A 'Building Flow' module exists within NAME to model flow around an isolated, cuboid-shaped building. It is intended to represent the perturbation of the background flow caused by the presence of the building, and to capture, to first order, effects such as building wakes and recirculation regions.



As a stand-alone model, the building model performs well. However further work is required to fully integrate its functionality into NAME. At the present time, some aspects of this integration might not be correct and could lead to strange behaviour. Current advice is that the building scheme should not be used until it has been more thoroughly tested.

1.6.4.3 The prototype flow module

A 'Prototype Flow' (associated with a type of met data referred to as 'prototype met') is available in NAME, although it is not normally used in practice. It was added during early development work on NAME. The prototype flow module provides an idealised flow field that is horizontally homogeneous and time independent and can be regarded as a simplified representation of a neutral boundary layer.

1.7 Physical Processes in NAME

NAME is capable of representing various processes involved in the release of pollutants and their subsequent transport, evolution and removal from the atmosphere. For instance, it can consider the initial rise of a plume from an industrial stack due to buoyancy or momentum effects, the chemical transformation of pollutants within the atmosphere and their removal to the ground surface by deposition processes. The user can select which processes are considered by NAME and control many aspects of the modelling behaviour. These global modelling options are generally specified via the [Sets of Dispersion Options](#) block in the main input file.

1.7.1 Specifying dispersion modelling options

Many characteristics of a NAME simulation can be configured by the user by specifying appropriate dispersion modelling options to be used in the [Sets of Dispersion Options](#) input block. These options include 'switches' to activate certain aspects of model functionality (such as the treatment of radiological decay, dry deposition, etc.) and also other parameters affecting the global behaviour of NAME (such as puff modelling options, time limits to apply to random-walk schemes, etc.). Table 1.24 summarises the various dispersion options that are available, with a more detailed discussion included in the following subsections.

There is usually only one set of dispersion options specified within a NAME input file, which is treated using a single 'case' within the NAME run. However multiple sets of dispersion options may also be given and, if so, these will be treated as multiple cases within the simulation (see [RUNNING MULTIPLE 'CASES'](#) in 1.2.7.5). When multiple sets of dispersion options are supplied in the input file, then the number of sets should be listed under the variable [Dispersion Options Ensemble Size](#) in the input block [Multiple Case Options](#). Multiple sets of dispersion options can be useful for exploring the influence of modelling choices on the results (e.g., looking at differences between a particle run and a puff run, or assessing the effect of using different random-walk schemes).

1.7.2 Running NAME with particles

The maximum permitted number of particles that can be represented at any stage in the NAME simulation needs to be specified using the [Max # Particles](#) keyword in the [Sets of Dispersion Options](#) input block. The value supplied here is used at the start of each dispersion case to allocate the necessary computer memory to store the particle information. A reasonably large value (say, one million particles) is often sufficient to cover many modelling applications, but the user will need to provide a bigger value for set ups requiring a large number of particles. As a general guideline, the value set should be slightly greater (say, by 20%) than the expected maximum number of particles that are likely to be realised in practice, so as to provide a buffer against running out of particles. Note that there is an overhead in allocating the memory to store particle information (in terms of

Keyword name	Description of dispersion option
Max # Particles	Maximum number of particles allowed in the simulation
Max # Full Particles	Maximum number of 'full' particles allowed in the simulation (that is, particles for which extra information is stored)
Max # Puffs	Maximum number of puffs allowed in the simulation
Max # Original Puffs	Maximum number of original puffs allowed in the simulation (that is, puffs released at a source and not created by puff splitting)
Particle Ceiling	Particle number ceiling above which the number of particles released and the mass per particle (but not the total mass released) are adjusted in an attempt to prevent the model from running out of available particles as the particle limit is approached
Particle Factor	Scale factor by which the mass per particle is increased when the particle number is above the Particle Ceiling value
Inhomogeneous Time	Travel time over which inhomogeneity is represented
Skew Time	Travel time over which skewness is represented
Velocity Memory Time	Travel time over which velocity memory is represented
Mesoscale Velocity Memory Time	Travel time over which velocity memory for unresolved mesoscale motions is represented
Puff Time	Travel time over which puffs are used
Puff Interval	Time interval between release of two consecutive computational puffs
DeltaOpt	Identifier of approach used to compute puff spread
A1	Parameter in puff scheme
A5	Parameter in puff scheme
A7	Parameter in puff scheme
Sync Time	Time interval at which particles and/or puffs are synchronised (this is effectively the main model time step)
Computational Domain	Name of the computational domain
Time of Fixed Met	Time of fixed met data (for a 'FixedMet' run)
Deep Convection?	Specifies scheme to use for representing deep convective mixing
Radioactive Decay?	Switch to indicate that radioactive decay is represented
Agent Decay?	Switch to indicate that agent decay is represented
Dry Deposition?	Switch to indicate that dry deposition is represented
Max Deposition Height	Height below which particles are subject to dry deposition
Wet Deposition?	Switch to indicate that wet deposition is represented
Turbulence?	Switch to indicate that turbulence is represented
Mesoscale Motions?	Switch to indicate that unresolved mesoscale motions are represented
Chemistry?	Switch to indicate that the chemistry scheme is run
EulerianModel	Switch to indicate that the Eulerian model is run
Damping?	Switch to indicate that near-source damping of eddy diffusive growth is represented
Vertical Velocity?	Switch to indicate that particles are advected in three dimensions (otherwise 2-d constant-height trajectories are generated)

Table 1.24: List of dispersion modelling options in NAME.

the RAM used by the NAME process), which the user should bear in mind especially when running large NAME jobs or running on systems with limited RAM.



To avoid confusion here, [Max # Particles](#) describes the maximum number of particles that can be stored *at any single instance* during the simulation and not the total number of particles considered throughout the model run (which can potentially become very large for extended runs). The difference here is due to an ability to 'recycle' particles during a NAME run, so that once a computational particle is no longer active (e.g. because it has left the computational domain) then the memory associated with that particle can be reused.

A secondary array containing additional particle information (referred to as 'Particle Extras') is also used when modelling plume rise or using the 'near-source' dispersion scheme. Memory is allocated for the 'Particle Extras' structure based on the value supplied for the [Max # Full Particles](#) variable in the [Sets of Dispersion Options](#) block. The [Max # Full Particles](#) value is usually much smaller than the corresponding [Max # Particles](#) value, because the extra information is only required for particles during the initial transport phase when plume-rise calculations are being performed or the 'near-source' scheme is active (which might typically extend to the initial 30 minutes or one hour after release). The exception here would be for purely short-range dispersion problems where the extra information might be required throughout the model simulation. In this case, the values for [Max # Particles](#) and [Max # Full Particles](#) should be set the same.

As with the maximum number of particles, there is a memory overhead when allocating the 'Particle Extras' structure and [Max # Full Particles](#) should not be set unnecessarily high when this aspect of the model functionality is not used (or is only applied to a small set of particles at any one time). When not used, it is advisable to set [Max # Full Particles](#) to the value [2](#) (in principle it should be possible to set the value to zero, however it is unclear whether all compilers would support allocating an array of size zero!).

The [Max # Particles](#) value indicates the maximum number of particles that can be represented at any instant during the model run. If this limit is actually reached at any time, then all subsequent particle releases (and splitting of any massive particles) will cease until some existing particles become inactive and the memory associated with them can be re-used. The situation where particle releases and/or particle splitting has to be deactivated is undesirable, as no source terms will be recognised during any such period. A method has therefore been introduced to reduce the risk of completely running out of available particles to release, whereby all particle release rates are universally reduced as the global particle limit is approached, see [CONTROLLING PARTICLE NUMBERS IN A NAME RUN](#) in 1.5.6.

The scheme uses the two keywords [Particle Ceiling](#) and [Particle Factor](#). The [Particle Ceiling](#) value defines the threshold number of particles beyond which release constraints are applied. When

the number of active particles exceeds this threshold, the mass *per particle* for any new particles being released (or split) is increased by the factor specified in [Particle Factor](#) and the release (or splitting) rate is reduced accordingly to retain the same total mass. So, for example, if [Particle Factor](#) is set to [2.0](#) and the [Particle Ceiling](#) threshold is exceeded, then any new particles that are subsequently released will have twice the usual mass of particles from that particular source but these will only be released half as frequently. This adjustment will continue up until a time when the number of active particles falls back below the threshold and the normal release or splitting criteria are resumed.

If [Particle Ceiling](#) and [Particle Factor](#) are not specified in the input file, then the default behaviour is to give no reduction in release rate (until the [Max # Particles](#) particle limit is reached at which point all releases will be forced to cease). When used, [Particle Ceiling](#) should be some value less than the maximum particle limit [Max # Particles](#) (at the 80% threshold, say), and [Particle Factor](#) should be a value greater than 1.

1.7.3 Running NAME with puffs

One of the main shortcomings of Lagrangian particle models is that they suffer from a problem of statistical noise. This arises out of their need to calculate concentrations by counting particles in grid boxes. In principle, the issue can be overcome by following a very large number of model particles, but this makes a model very expensive to run as the noise decreases only very slowly (behaving as the inverse of the square root of the number of particles). Grid box averaging also smoothes the concentration field, leading to underprediction of peak concentration magnitudes and overprediction of the area affected. Averaging widens the plume, and this effect can be large when the plume is narrow relative to the scale of the grid boxes.

To address these problems, NAME incorporates a puff model intended for short-range dispersion applications. Here each puff is designed to represent some of the material spread intrinsically. The puff scheme aims to reproduce the results of the particle model with reasonable accuracy whilst reducing the noise and computational cost encountered with the pure particle approach. The technique uses multiple puffs, with some of the dispersion being represented by the growth of these individual puffs and some by the random motion of the puff centres as in a particle model. The approach has similarities with that of de Haan and Rotach [1998], although the puffs in NAME are constructed in a numerical sense rather than having any physical form or significance. That is, they are designed as a mathematical construct to obtain a good approximation to the true solution of the stochastic Lagrangian particle model.¹⁰

The puff scheme in NAME introduces smoothing by replacing each individual point-mass particle by a distributed puff centred on the particle location. The tendency of this procedure to systematically widen the plume and reduce concentrations is compensated for, at least approximately, by

¹⁰The true solution being only obtainable, in general, in the limit of a simulation with infinitely many particles.

reducing the spread of the puff centres. In this way, the approach is explicitly designed to avoid artificial widening of the plume. The partitioning between the spread represented through the growth of individual puffs and that modelled through the random motion of the puff centres is tuneable. At one extreme the puff scheme can have a very large number of very small puffs (which approaches the accuracy of a pure particle model), while at the other extreme the puff approach can treat a few, very large puffs (which gives a faster model with accuracy comparable to an ensemble mean approach to the dispersion). In other words, the model is tuneable to give the desired balance between speed and accuracy.

Puffs are also given a spread in time, so that they contribute to concentrations at times either side of their nominal time. This approximation can be made because the meteorology usually changes on a much slower time scale than the puff release interval (which might typically be chosen to be a few minutes, say) and so any further puff released within this time spread would behave in a very similar manner to the original puff. To ensure that results vary smoothly, the time-spread of a puff is taken to be triangular in shape (apart from the first and last puffs from a source) with an overlap in time between any two successive puff releases. Note that the puff model also splits or recombines puffs, as required, to ensure sufficient puffs are present to adequately represent the concentration field at any moment in time.

Output quantities computed from puffs use the spot value at a grid point (rather than calculating a grid box average as with the particle approach). In fact, spatial averaging of puffs over a grid box is not currently supported by NAME, although this ability would be useful and may be added in the future.

The puff scheme, when activated, is applied to model the dispersion of material for an initial period of time following its release from a source. At the end of that time period, the puff is converted back into a standard particle with all its mass concentrated at the particle position (which, at the time of conversion, is represented by the puff centroid). The puff-to-particle transition time is specified using the [Puff Time](#) variable in the [Sets of Dispersion Options](#) input block, and must be a non-negative time interval. A zero time interval `00:00` indicates that the puff scheme is never invoked and the dispersion is represented using particles throughout. When the user supplies a positive time interval here, emissions are initially represented using puffs which then transition to particles after the specified travel time has elapsed.

The two variables [Max # Puffs](#) and [Max # Original Puffs](#) in the [Sets of Dispersion Options](#) input block must be given suitable (positive) values when puffs are being used. The keyword [Max # Puffs](#) specifies the maximum permitted number of puffs that can be represented in the NAME simulation at any moment in time (allowing for recycling of puffs in the same way that the particles can be recycled). The keyword [Max # Original Puffs](#) indicates the maximum number of primary puffs that can be released from the sources (again after allowing for the recycling of puffs). Note that the total number of puffs within the simulation is expected to be greater than the number of primary puffs released from sources due to the puff splitting. The values supplied in these two variables

are used at the start of each dispersion case to allocate the necessary computer memory to store the puff information. Reasonably large values (say, one hundred thousand puffs and one thousand puffs, respectively) are usually sufficient to cover most modelling scenarios, but the user will need to provide bigger values for set ups requiring a large number of puffs to be modelled. As a general guideline, the values set should be slightly greater (say, by 20%) than the numbers that are likely to be realised in practice, so as to provide a buffer against running out of puffs. Note that there is an overhead in allocating the memory to store puff information (in terms of the RAM used by the NAME process), which the user should bear in mind especially when running large NAME jobs or running on limited-RAM systems. In particular, the user should try to avoid setting unnecessarily high values and, when the puff scheme is not being used (that is, **Puff Time** is set as **00:00**), then it is good practice to set each of these values to **2** to minimise memory use (in principle it should be possible to set the values to zero, however it is unclear whether all compilers would support allocating an array of size zero!).

When using the puff scheme, the user must also specify a non-zero time interval for the keyword **Puff Interval** in the **Sets of Dispersion Options** block. This sets the time interval between successive computational puffs released by a source (or equivalently the time half-spread of each computational puff). Each computational puff is assumed to be representative of any puff released within its time spread, an approximation that can be made because the meteorology usually changes on a much slower time scale than the puff release interval. In practice the **Puff Interval** variable is chosen to be of the order of one to five minutes.

There are several further variables related to the puff scheme that can be controlled by the user via keywords in the **Sets of Dispersion Options** input block. These govern various aspects of the puff calculations (e.g. limits on the sizes of individual puffs which regulates the degree of puff splitting that is allowed to occur during the simulation). Variables that can be specified in this way via the input file are **DeltaOpt** (which is usually set to **1**) and the three parameters **A1** (default: **50.0**), **A5** (default: **3.0**) and **A7** (default: **2.0**). Use of the default values is generally recommended. The **DeltaOpt** variable must be specified in the input file, but the three A-parameters can be left blank (or omitted entirely) to use the above defaults. Users interested in investigating these puff options further should contact us for advice.



The puff scheme is a research option and is not compatible with all modelling options in NAME. In particular, puffs cannot be used when modelling chemistry.

For further information on the puff model in NAME, see Thomson and Jones [2011], and also the document >> **MD1/1: DISPERSION MODELLING – PARTICLES, PUFFS, KERNELS AND BOX AVERAGING** <<.

1.7.4 Advection and diffusion

1.7.4.1 Introduction

Particles are advected in three dimensions in the model by a mean wind (e.g. resolved winds from UM) interpolated to the particle position and time, with unresolved motions such as turbulence being simulated by random walk techniques of various levels of sophistication. The general form of the equation of motion used in NAME is given by

$$\mathbf{x}_{t+\Delta t} = \mathbf{x}_t + (\bar{\mathbf{u}}(\mathbf{x}_t) + \mathbf{u}'(\mathbf{x}_t) + \mathbf{u}'_m(\mathbf{x}_t)) \Delta t \quad (1.7.1)$$

where the change in the position of a particle, \mathbf{x} , from its position at time t over a time interval Δt is expressed using the sum of three separate velocity terms: the mean wind $\bar{\mathbf{u}}$ at the particle position and turbulent velocities \mathbf{u}' and \mathbf{u}'_m arising from separate treatment of the turbulence scales and unresolved mesoscale motions, respectively, within NAME.

A description and basic formulation of each scheme is presented in the section [INTRODUCING THE RANDOM-WALK SCHEMES](#) in 1.7.4.2 below. The following sections ([TURBULENCE](#) in 1.7.4.3 and [UNRESOLVED MESOSCALE MOTIONS](#) in 1.7.4.4) then describe the parametrizations used in NAME for estimating the turbulence and unresolved mesoscale motion variables required by these random-walk schemes. In particular, NAME requires vertical profiles of the velocity variances and (de)correlation timescales. When modelling turbulence within the atmospheric boundary layer, both inhomogeneous and homogeneous (boundary-layer average) formulations are available. Separate formulae are used for stable and unstable conditions. In the unstable case, both Gaussian and non-Gaussian (skewed) turbulence schemes are available. Fixed turbulence values are applied in the free troposphere above the atmospheric boundary layer (although the defaults for these fixed values can be modified in the user input). Single fixed (met data resolution dependent) values are also used when modelling unresolved mesoscale motions, and again these values can be modified by the user if necessary. The final section [USING THE RANDOM-WALK SCHEMES](#) in 1.7.4.5 then discusses how these random-walk schemes are implemented and controlled in practice within NAME.

1.7.4.2 Introducing the random-walk schemes

The dispersion of air pollutants by small-scale three-dimensional atmospheric turbulence and unresolved mesoscale motions are parametrized within NAME using random-walk techniques. The aim of the random-walk (dispersion) model is to compute an ensemble of random trajectories of marked fluid (or Lagrangian) particles through a flow field whose statistics are known. Random-walk methods have been used for modelling atmospheric dispersion for at least three decades. During that time the theoretical basis of these models has been established and their range of applications has become very large (for a discussion of both the history and applications of these models see, e.g., Thomson and Wilson [2013]). The computational power available to such models has increased

enormously over this period so that problems that were hitherto impossible are now routine and this has enabled models of this type to be extensively validated by comparison with observations.

This section introduces the two different random-walk models that are used in NAME to represent the dispersion and explains why they are appropriate models for use in realistic atmospheric situations.

The long-range (diffusive) scheme

The simplest example of a random-walk model is a diffusion process (also known as standard Brownian motion). Formally it is a *Wiener process*, which is discussed in detail in standard textbooks on stochastic processes, such as [Gardiner, 1985, p.66]. In the context of atmospheric dispersion, the diffusion of a Lagrangian particle in this case is specified by a diffusivity, $\mathbf{K} = (K_x, K_y, K_z)$, which is related to the turbulent velocities and time scales of atmospheric motions. In NAME, the along-wind and cross-wind spread are assumed to be equal, and the eddy diffusivity is further assumed to take the form

$$\mathbf{K} = (\sigma_u^2 \tau_u, \sigma_u^2 \tau_u, \sigma_w^2 \tau_w), \quad (1.7.2)$$

where σ_u and σ_w are the standard deviations of the horizontal and vertical velocity fluctuations, respectively, and τ_u and τ_w are the corresponding horizontal and vertical Lagrangian timescales. Details of the precise form of these quantities are described in the following two sections [TURBULENCE](#) (see 1.7.4.3) and [UNRESOLVED MESOSCALE MOTIONS](#) (see 1.7.4.4). Turbulence profiles in NAME are usually taken to be constant with height within the boundary layer (homogeneous turbulence) when using this diffusive random walk model, but alternatively they may vary with height (inhomogeneous turbulence).

The diffusive random walk is given by the stochastic differential equation (in its discretised form)

$$\mathbf{x}(t + \Delta t) = \mathbf{x}(t) + \bar{\mathbf{u}}\Delta t + \sqrt{2\mathbf{K}\Delta t} \mathbf{r} \quad (1.7.3)$$

where \mathbf{r} is a random variable drawn from a normal distribution with zero mean and unit variance. In this case, the advection of the particle is due to the mean wind $\bar{\mathbf{u}}$ (supplied through the input meteorology) together with random displacements representing the velocity components due to turbulence and unresolved mesoscale motions that have no memory from one time step to the next. A form of 1.7.3 is implemented in NAME, but using a damped eddy diffusivity term at small travel times to approximate the correct dispersion behaviour near to a source (see Figure 1.9).

The short-range (velocity-memory) scheme

A more complete description of the dispersion is obtained by including a ‘memory’ for the particle velocity. The implementation in NAME of this more extensive treatment is based on the Langevin equation for homogeneous turbulence but also accounts for the influence of inhomogeneous turbulence. The Langevin equation is first introduced and discussed, and then its generalisation to inhomogeneous turbulence as implemented in NAME is considered.

In its most general form, the stochastic differential equation that governs the evolution of the

velocity $\mathbf{u} = (u_1, u_2, u_3)$ of a dispersing particle is given by

$$du_i = a_i(\mathbf{x}, \mathbf{u}, t) dt + \sum_{j=1}^3 b_{ij}(\mathbf{x}, \mathbf{u}, t) dW_j, \quad i = 1, 2, 3; \quad (1.7.4)$$

where dW_j is a random Gaussian increment with zero mean and variance dt , and a and b are to be determined. When the turbulence is homogeneous (that is, the velocity variances do not depend on position), it can be shown that 1.7.4 reduces to the *Langevin equation* [Gardiner, 1985, p.80], viz.

$$du_i = -\frac{u_i}{\tau_i} dt + \sqrt{\frac{2\sigma_i^2}{\tau_i}} dW_i, \quad i = 1, 2, 3; \quad (1.7.5)$$

where $\sigma^2 = (\sigma_u^2, \sigma_v^2, \sigma_w^2) = (\sigma_1^2, \sigma_2^2, \sigma_3^2)$ is the velocity variance and $\tau = (\tau_u, \tau_v, \tau_w) = (\tau_1, \tau_2, \tau_3)$ is the time scale over which the velocity \mathbf{u} decorrelates. Formally, a stochastic process of this form is known as an *Ornstein-Uhlenbeck process*, [Gardiner, 1985, p.74]. It is also known as a *mean-reverting process* because the damping term (the first term on the right-hand side of 1.7.5) ensures that the velocity reverts to its large-time mean on the time scale given by τ . For example, if the vertical velocity w is negative then the drift term will be positive and vice-versa. It is for these reasons that the velocity in 1.7.5 can be thought of as having a ‘memory’. Both the Wiener process and the Ornstein-Uhlenbeck process are examples of Markov processes because the variables involved only depend on their values at the previous time step and not on their earlier history.

The model 1.7.5 is constructed to be consistent with Kolmogorov’s hypothesis for homogeneous isotropic turbulence, which constrains the form of b_{ij} in terms of the mean kinetic energy dissipation rate ε and constant C_0 (see, e.g., [Rodean, 1996, p.10] and [Pope, 2000, p.486]). These parameters are estimated in NAME from the input meteorological variables, see the next section [TURBULENCE](#) in 1.7.4.3 for details.

The motivation for using an Ornstein-Uhlenbeck process to model the near-source dispersion is that it correctly captures the spread of a cloud of particles from a point source. This was demonstrated for homogeneous turbulence in a famous paper by Taylor in 1922, see Taylor [1922]. It can be shown that when modelled by a Wiener process, the spread of a cloud of particles from a point source is parabolic in form, whereas the initial spread modelled by an Ornstein-Uhlenbeck process is linear (Figure 1.9) and agrees with observations. For travel times much larger than τ , the spread computed from an Ornstein-Uhlenbeck process is also parabolic. This demonstrates that if the primary interest is in long-range dispersion, it is sufficient to use a Wiener process. The details of this analysis can be found in Taylor [1922] and in standard textbooks, e.g., [Pal Arya, 1999, p.155] or [Pasquill and Smith, 1983, p.115].

As the velocity-memory approach is intended for use in the atmospheric boundary layer where the turbulence is not homogeneous, the option of using the velocity-memory model 1.7.5 with homogeneous turbulence profiles is not allowed for in NAME. The extension of 1.7.5 to include inhomogeneous turbulence is not trivial because there is a net acceleration of the form $\partial\sigma_w^2/\partial z$ (since σ_w^2 is

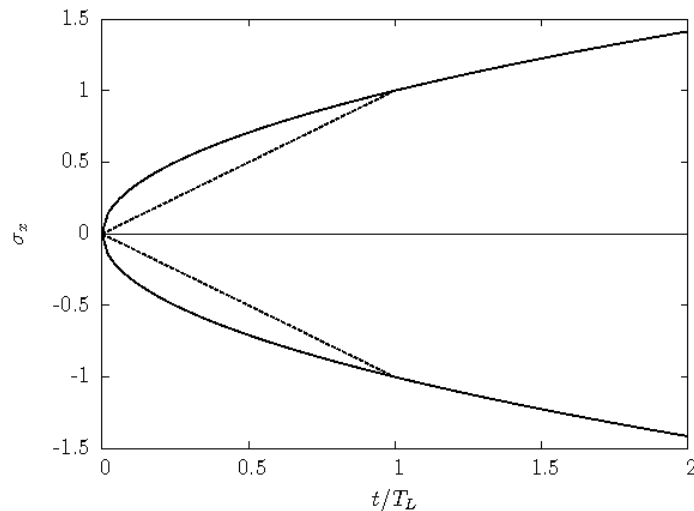


Figure 1.9: Conceptual evolution of the spread σ_x of a cloud of particles from a point source as a function of travel time, as modelled by a Wiener process (solid line) and an Ornstein-Uhlenbeck process (dotted line).

not constant in the vertical for inhomogeneous turbulence). This needs to be properly accounted for to ensure that there is no spurious clustering of particles in regions where the velocity variances are small. In the atmospheric context, where the velocity variances are functions of height, this would result in particles clustering around the top of the boundary layer (since σ_w decreases with height, it is easier for particles to move upwards than downwards). Thus, there is a drift of particles from regions of relatively high turbulent velocity to regions of relatively low turbulent velocity.

A generalisation of 1.7.5 to include inhomogeneous turbulence can be achieved rigorously by means of the *well-mixed condition* of Thomson [1987]. The well-mixed condition provides a constraint on the form that Lagrangian stochastic models can take. It requires an initially uniform distribution of particles to remain uniform throughout its evolution. Since the particles follow the fluid exactly, incompressibility ensures and requires that a cloud of particles cannot ‘unmix’ itself. This should remain the case even if the turbulence is inhomogeneous. As the Eulerian statistics are assumed to be known, they may be used to constrain the form of the Lagrangian stochastic model and this provides a method of determining the form of the model. The drift term (the first term on the right-hand side of 1.7.4) takes the form

$$a_i = -\frac{1}{2}C_0\varepsilon \sum_{j=1}^3 (V^{-1})_{ij} u_j + \frac{1}{2} \sum_{j=1}^3 \sum_{k=1}^3 (V^{-1})_{kj} \frac{dV_{ik}}{dt} u_j, \quad i = 1, 2, 3;$$

where V is the velocity covariance matrix. For details of this derivation, see [Rodean, 1996, Chps 7, 8]. It is important to note that the well-mixed condition does not constrain the model uniquely in more than one dimension. The generalisation of the Langevin equation to include the influence of inhomogeneous turbulence is sometimes referred to in NAME as a *Langevin-type approach*.

The implementation in NAME of this more extensive treatment based on the Langevin equa-

tion (but which also accounts for the influence of inhomogeneous turbulence) is referred to as the *velocity-memory scheme*. In principle, the velocity-memory approach could be used with homogeneous turbulence (although as we have seen there are reasons why this combination is not allowed for in NAME). The combination of the velocity-memory scheme with inhomogeneous turbulence profiles is called the *near-source scheme* as it is intended for use at small travel times.

1.7.4.3 Turbulence

This section now looks at how NAME estimates turbulence parameters for the random-walk schemes described in [RANDOM-WALK SCHEMES](#) in 1.7.4.2. Inhomogeneous turbulence profiles within the boundary layer have been derived from published empirical fits to observational data and large eddy simulations (LES). These vertical profiles of turbulence parameters depend on the stability of the atmosphere, which is determined by the sign of the calculated Monin-Obukhov length (L). The inhomogeneous profiles are functions of height and, by averaging over the boundary layer, homogeneous profiles can be constructed as boundary-layer mean values of the inhomogeneous profiles. Note that turbulence parameters can vary with horizontal position (for example, when using NWP meteorology) and with time. Turbulence parameters are assumed to be horizontally isotropic in NAME, as currently there is no distinction between along-wind and cross-wind turbulence parameters. Since lateral dispersion is considered more important than along-wind dispersion, horizontal turbulence parameters are intended to be representative of the cross-wind dispersion.

In stable conditions ($L \geq 0$), the inhomogeneous velocity variance profiles ($\sigma_{u,v,w}^2$) are given by

$$\begin{aligned}\sigma_u^2 (= \sigma_v^2) &= 4.0u_*^2 \left(1 - \frac{z}{z_i}\right)^{3/2}, \\ \sigma_w^2 &= 1.69u_*^2 \left(1 - \frac{z}{z_i}\right)^{3/2},\end{aligned}$$

where z is the height above ground, z_i is the boundary layer depth and u_* is the friction velocity. The meteorological variables are given by, or determined from, the input meteorological data. In unstable conditions ($L < 0$), the inhomogeneous velocity variable profiles are given by

$$\begin{aligned}\sigma_u^2 (= \sigma_v^2) &= 0.4w_*^2 + 4.0u_*^2 \left(1 - \frac{z}{z_i}\right)^{3/2}, \\ \sigma_w^2 &= 1.2w_*^2 \left(\frac{z}{z_i}\right)^{2/3} \left(1 - \frac{z}{z_i}\right) + 1.69u_*^2 \left(1 - \frac{z}{z_i}\right)^{3/2},\end{aligned}$$

where w_* is a characteristic convective velocity defined by

$$w_* = u_* \left(\frac{z_i}{\kappa|L|}\right)^{1/3}$$

where κ is von Kármán's constant (see Table 1.2).

The homogeneous velocity variance profiles are given by the boundary layer mean values of the

inhomogeneous profiles. In stable conditions, the homogeneous profiles are given by

$$\begin{aligned}\sigma_u^2 (= \sigma_v^2) &= 1.6u_*^2, \\ \sigma_w^2 &= 0.676u_*^2.\end{aligned}$$

Whereas, in unstable conditions the homogeneous velocity variance profiles are given by

$$\begin{aligned}\sigma_u^2 (= \sigma_v^2) &= 0.4w_*^2 + 1.6u_*^2, \\ \sigma_w^2 &= 0.27w_*^2 + 0.676u_*^2.\end{aligned}$$

The inhomogeneous profile of the dissipation rate of turbulent kinetic energy (ϵ) is given in stable conditions by

$$\epsilon = \frac{u_*^3}{k} \left(\frac{1}{z} + \frac{4}{L} \right) \left(1 - \frac{z}{z_i} \right),$$

and in unstable conditions by

$$\epsilon = \left[1.5 - 1.2 \left(\frac{z}{z_i} \right)^{1/3} \right] \frac{w_*^3}{z_i} + \frac{u_*^3}{kz} \left(1 - \frac{z}{z_i} \right).$$

The Lagrangian timescales ($\tau_{u,v,w}$) are calculated from the velocity variances and the rates of dissipation of turbulent kinetic energy,

$$\tau_{u,v,w} = \frac{2\sigma_{u,v,w}^2}{C_0\epsilon},$$

where in stable conditions $C_0 = 5.0$, and in unstable conditions

$$C_0 = 2.1 \log(3|L|) - 1.5,$$

with the constraints

$$\begin{aligned}C_0 &\geq 1.0 && \text{unstable limit,} \\ C_0 &\leq 5.0 && \text{neutral limit.}\end{aligned}$$

The homogeneous Lagrangian timescales and dissipation rates of turbulent kinetic energy are given by the boundary layer means of the inhomogeneous values (see Webster et al. [2003]).

In Gaussian turbulence the probability that a sampled vertical velocity will be positive is the same as the probability that it will be negative. In other words, it is just as likely that an air parcel will be moving upwards as it is downwards. In skewed turbulence these probabilities are not equal. Turbulence in convective conditions is, in general, skewed with relatively strong updrafts within narrow buoyant plumes counterbalanced by weaker downdrafts over a wider region. NAME can represent this skewed turbulence as the sum of two Gaussian functions – one that represents the updrafts and the other for the downdrafts. Details appear in Maryon [1997] and Maryon et al. [1999].

When gravitational settling of heavy particles is being modelled, the NAME computational particles are not carried inertly along with the flow but instead have a tendency to fall through the turbulent eddies. Here the turbulence parameters are modified by the sedimentation velocity w_T to account for trajectory-crossing and inertial effects (see Apsley [1989]). Specifically, the Lagrangian timescales and turbulent velocity variances are reduced according to the formulae

$$\tau_{u,v,w}^* = \frac{\tau_{u,v,w}}{\sqrt{1 + w_T^2/\sigma_w^2}}, \quad \sigma_{u,v,w}^* = \frac{\sigma_{u,v,w}}{\sqrt{1 + T_p/\tau_{u,v,w}^*}}$$

where T_p is the particle time constant

$$T_p = \frac{w_T}{g} \frac{\rho_p - \rho}{\rho_p} \approx \frac{w_T}{g}.$$

In the free troposphere¹¹ (the region above the boundary layer), turbulence parameters are assumed to be fixed. Default values are $\sigma_u = \sigma_v = 0.25 \text{ m s}^{-1}$, $\sigma_w = 0.1 \text{ m s}^{-1}$, $\tau_u = \tau_v = 300 \text{ s}$ and $\tau_w = 100 \text{ s}$. The user can modify the fixed free tropospheric turbulence parameters used with single-site meteorological data or NWP meteorological data by specifying alternative values in the variables [Free Trop SigU](#), [Free Trop SigW](#), [Free Trop TauU](#) and [Free Trop TauW](#) in the appropriate met module instances block ([Single Site Met Module Instances](#) or [NWP Met Module Instances](#)).

1.7.4.4 Unresolved mesoscale motions

Low frequency horizontal eddies with scales which lie between the resolved motions of the input meteorological data and the small three-dimensional turbulent motions covered by the turbulence parametrizations are parametrized separately in NAME by the unresolved mesoscale motions scheme.

The unresolved mesoscale motions parameters (velocity variances and Lagrangian timescales) are dependent on the input meteorological data, although these default values may be changed by the user. When not provided as user input, default values are presented in Table 1.19 when using NWP met data (see Webster and Thomson [2015]), and $\sigma_u = \sigma_v = 0.5 \text{ m s}^{-1}$ and $\tau_u = \tau_v = 7200 \text{ s}$ when using single site met data. Here the differences between these default values are explained by the different spectral characteristics of each type of input meteorological data and the timescales that are being captured explicitly by the met data. Single-site meteorology consists of observations at an hourly frequency and the choice of the unresolved mesoscale motion parameters is intended to represent the low-frequency meandering eddies that are not resolved by an hourly data set. For NWP data sets, the unresolved mesoscale motions parameters should reflect the scales of motion that are resolved by the NWP model (and can be determined formally by examining the ‘missing energy’ between the model and observed wind spectra). In principle, higher resolution

¹¹The free troposphere is the layer within the troposphere above the planetary boundary layer. The region is typically characterised by laminar flow, although some turbulent regions may exist especially in areas of large-scale convective activity or breaking gravity waves.

models (which can resolve more of the smaller scales of motion) require smaller values for the unresolved mesoscale motions parameters because fewer of the atmospheric motions at these scales need to be parametrized. Values will also depend on other characteristics of the NWP modelling system. The user can modify the unresolved mesoscale motions parameters used with single site meteorological data or NWP meteorological data by setting the variables [Mesoscale SigU](#) and [Mesoscale TauU](#) in the appropriate met module instances block ([Single Site Met Module Instances](#) or [NWP Met Module Instances](#), respectively).

1.7.4.5 Using the random-walk schemes

The random walk techniques used in NAME have been introduced in [RANDOM-WALK SCHEMES](#) in 1.7.4.2. At short ranges, a Langevin-type approach in which the turbulent velocities are correlated in time (commonly referred to as ‘velocity memory’) can be used, whilst at longer ranges a simpler, less expensive Wiener (diffusive) process is applied. In both approaches, the random turbulent velocity components are functions of turbulence parameters (velocity variances and Lagrangian timescales or turbulence dissipation rates). The present section now considers how these schemes are invoked in practice within NAME.

Within the NAME input file, the keyword [Turbulence?](#) in the [Sets of Dispersion Options](#) block acts as a global switch for the turbulence parametrization. Normally [Turbulence?](#) is set to [Yes](#) to include representation of the turbulence in the NAME modelling of particle trajectories. Both homogeneous (constant with height within the atmospheric boundary layer) and inhomogeneous (functions of height within the atmospheric boundary layer) profiles of turbulence parameters are available for use (see setting [Inhomogeneous Time](#) below).

When using the near-source scheme, it is not necessary to explicitly prescribe a shorter sync step as the scheme automatically calculates an internal short time step for the particle calculations (though there may be other reasons to use a shorter sync time if the focus of the NAME application is on modelling the short range). The scheme is computationally more expensive than the Wiener approach, and so is often not invoked or, when it is, it is only usually applied near to a source before switching to the simpler, computationally cheaper Wiener process at longer range. The near-source scheme is invoked by setting non-zero time intervals for the keyword variables [Velocity Memory Time](#) and [Inhomogeneous Time](#) in the [Sets of Dispersion Options](#) block of the main input file. Each particle released will then use the near-source turbulence scheme for the period of time specified (typically 30 minutes to one hour) before switching to the simpler and cheaper longer range turbulence scheme. It is usual to specify the same time interval for the two variables [Velocity Memory Time](#) and [Inhomogeneous Time](#) but specifying a shorter period for [Velocity Memory Time](#) than that specified for [Inhomogeneous Time](#) (so as to invoke the use of inhomogeneous turbulence parameters without velocity memory) is allowed (see below). Conversely, the opposite selection (i.e., use of less accurate homogeneous (mean) turbulence parameters with velocity memory) is not permitted.

Turbulence in the basic near-source scheme is assumed to be Gaussian (and, in particular, the probability that a sampled vertical velocity will be positive is the same as the probability that it will be negative). However, a 'skewed' turbulence scheme also exists which can be invoked with the near-source scheme (i.e., with velocity memory and with inhomogeneous turbulence parameters) and which is designed to be used in convective conditions when turbulence is, in general, non-Gaussian with narrow plumes of relatively strong updraughts and wider areas of weaker downdraughts. The skewed turbulence scheme is invoked by specifying a non-zero time interval for the variable **Skew Time** in the **Set of Dispersion Options** block. Each particle will then use a skewed turbulence scheme for the initial **Skew Time** period after release, before switching to a Gaussian turbulence scheme. The time interval given for the variable **Skew Time** must not be longer than that given for the variable **Velocity Memory Time**. In general, the following constraint holds between the various time intervals involved in the representation of turbulence: $\text{Skew Time} \leq \text{Velocity Memory Time} \leq \text{Inhomogeneous Time}$.

The combination of the simpler Wiener process and homogeneous turbulence parameters is often referred to as the 'long-range scheme'. It uses a relatively long timestep and is therefore comparatively cheap to run. There is a discontinuity in turbulence parameters at the boundary layer top, which is dealt with using an entrainment scheme (see Webster and Thomson [2011]). To use the long-range scheme throughout a NAME run, the variables **Skew Time**, **Velocity Memory Time** and **Inhomogeneous Time** should all be set to a zero time interval (that is, **00:00**). It is recommended that the keyword **Damping?** in the **Sets of Dispersion Options** block is set to **Yes** when the long range scheme is being used throughout (this is the default setting if not specified explicitly by the user). This setting ensures that the eddy diffusivity growth is appropriately damped near to a source (that is, at small travel times).

Inhomogeneous turbulence parameters can be used with the Wiener process, although this combination is not commonly used. There is no discontinuity at the boundary layer top with the inhomogeneous profiles so an entrainment scheme is not required here.

Unresolved mesoscale motions (i.e., eddies with scales lying between the resolved motions captured by the input meteorological data and the three-dimensional turbulent motions) are parametrized separately from the turbulence in NAME by the unresolved mesoscale motions scheme. The modelling of unresolved mesoscale motions is controlled by the variable **Mesoscale Motions?** in the **Sets of Dispersion Options** block of the main input file, which acts as a global switch for the unresolved mesoscale motions parametrization. Usually **Mesoscale Motions?** is set to **Yes**.

As in the case of the turbulence parametrization, the unresolved mesoscale motions parametrization uses random walk techniques, and again both a velocity memory scheme and a simpler Wiener process exist. The velocity memory scheme, although not often used for unresolved mesoscale motions, is invoked by specifying a non-zero time interval for the variable **Mesoscale Velocity Memory Time** in the **Sets of Dispersion Options** block. Each NAME particle will then use the unresolved mesoscale motions velocity memory scheme for this specified time period after release, before re-

verting to the simpler Wiener process scheme. The simpler Wiener process can be used throughout the NAME model run by specifying a zero time interval (*00:00*) for the variable [Mesoscale Velocity Memory Time](#). The eddy diffusivity growth near to source will be damped if the variable [Damping?](#) is set to [Yes](#) in the [Sets of Dispersion Options](#) block.

1.7.5 Vertical mixing by deep convective clouds

Atmospheric convection plays an important role in the vertical transport and mixing of airborne pollutants. Mixing within a convective boundary layer is represented as part of the turbulence parametrization scheme (see [TURBULENCE SCHEMES USED IN NAME](#) in 1.7.4.3). The current section is concerned with mixing associated with ‘deep’ moist convection within the troposphere. Convective clouds are characterised by a large vertical development, and their vertical extension can be comparable to that of the troposphere. As their horizontal length scale is of the order of a few kilometres and is sub-grid in large-scale NWP models of the atmosphere, convection needs to be parametrized [Emanuel, 1994].¹² Two parametrization schemes are available in NAME for representing the effect of vertical mixing of particles due to tropospheric convection. These are referred to as the *old* scheme (or original scheme) and the *new* scheme.

In the old deep convection scheme, described in Maryon et al. [1999], whenever a convective cloud is diagnosed based on the input meteorological data (usually from the Met Office Unified Model), particles between the cloud base and cloud top are randomly redistributed with a probability of 1/25-th of the NWP-diagnosed convective cloud fraction.

In the new scheme, a more sophisticated approach using mass fluxes is applied to estimate the amount of material transported inside the convective cloud. Here the enhanced vertical particle mixing caused by convection is represented in a stochastic way, using mass fluxes to calculate how many particles entrain to the cloud and, once entrained, how many move upwards and detrain to the environment. The updraught mass fluxes are parametrized using empirical formulae, based on results from Cloud Resolving Models, to estimate the cloud base mass flux from the convective precipitation rate. A subsidence flux, equal to the updraught flux, is applied to all particles to compensate for the upward motion. The scheme satisfactorily fulfils the ‘well mixed condition’ [Thomson, 1987]. See Meneguz and Thomson [2014] for further details.

When considering whether to run NAME with either convection scheme, attention must be paid to the resolution of the input NWP met data and whether convection is explicitly represented by the NWP model or not. The parametrizations are designed to work with global met data and/or met data with a horizontal resolution that is insufficient to permit the convection to be captured explicitly.

¹²Operational NWP systems have, in recent years, started to be run at kilometre-scale resolutions that explicitly resolve the (larger scales of) the atmospheric convection processes. Models run at such resolutions are sometimes referred to as ‘convection-permitting’ models, recognising the fact that they permit at least some of the convection to be represented through the explicit model dynamics.



The option to run NAME with the new convection scheme in **Backwards** mode is currently disabled because the design of the scheme does not yet support this feature, although it is intended to address this limitation in a future release.

If the user chooses to run with a convection scheme turned on, then it is often useful to request output of the meteorological fields *Convective Cloud Base* and *Convective Cloud Top* (under an **Output Requirements - Fields** block). These diagnostics have been made available as outputs in version 6.4. The values are returned in *m agl* and can give an indication of the vertical extent of the convective cloud diagnosed at a given location of interest, which can in turn be revealing of the strength of convection. The user can also request *Precipitation rate (mm/hr)* as an output, however should be aware that this is a total precipitation rate combining both the dynamic and convective precipitation components (convective precipitation will be made available as a separate output in a future release of NAME). To run the new convection scheme with NWP met data other than those provided by the UM, users are advised to ensure that convective precipitation rate (in mm/hr) and the relevant convective cloud diagnostics (convective cloud base and cloud top) are available as input fields to NAME, as the new convection scheme will fail to run if these parameters are absent from the NWP met files.

The convective mixing scheme in NAME is controlled through the keyword *Deep Convection?* in the **Sets of Dispersion Options** input block. This keyword must be specified by the user in the input file (i.e. it cannot be left blank) and should take one of the values *No*, *Old* or *New*. Simulations without any deep convective scheme are specified by setting the value of the *Deep Convection?* keyword to *No*. For simulations which include the representation of deep convection, the user has to make a choice between the original NAME scheme and the new scheme, and these are activated by setting the keyword to *Old* or *New*, respectively.



In versions of NAME prior to v6.4, only the original convective mixing scheme was available and the *Deep Convection?* keyword was a logical flag with the value *Yes* or *No* (or could be kept blank, which was equivalent to *No*). This behaviour has been changed at v6.4 and the options to specify *Yes* or to leave blank have been removed and any NAME run using either of these options will now fail with an error message.

1.7.6 Plume rise scheme for buoyant and momentum-driven releases

A plume rise scheme is available in NAME to model the initial rise of a plume due to buoyancy and momentum effects, until the plume can be considered to be passive. The scheme is based upon conservation equations of mass, momentum and heat and is designed to be used to model plumes from power station stacks and similar industrial releases. Whilst the plume rise scheme can be

implemented for other types of buoyancy- and momentum-driven releases (e.g., fires), no guarantee of its accuracy can be given since implicit assumptions in the plume rise scheme (such as top hat profiles of plume cross-sectional properties, neglecting the size of the source in comparison to the plume rise) may be violated. For further details on the NAME plume rise scheme, see Webster and Thomson [2002].

Implementation of the plume rise scheme is controlled *on a per-source basis* by the variable **Plume Rise?** in each **Sources** block in the input files. Additionally, the temperature of the effluent, **Temperature**, (in Kelvin) and either the volume flow rate, **Volume Flow Rate**, (in $\text{m}^3 \text{s}^{-1}$) or the exit velocity, **Flow Velocity**, (in m s^{-1}) need to be specified. Note that unless the plume rise scheme is explicitly invoked for a particular source, by setting the variable **Plume Rise?** to **Yes**, these additional variables are ignored. Only two-dimensional circular sources can be modelled under the plume rise scheme. Hence whenever **Plume Rise?** is set to **Yes** for a source, then **Shape** should be set to **Ellipsoid**, **dX** and **dY** should be set equal (and given in metres) and **dZ** should be set to **0.0**.

The plume rise scheme is applied on a particle basis from the time of release of a particle until the time that particle can be considered to be passive. The near-source scheme should be invoked during the plume rise period to ensure that an appropriately small time step is used for the plume rise calculations. Details of how to invoke the near-source scheme can be found in [USING THE RANDOM-WALK SCHEMES](#) in 1.7.4.5. Note that the NAME plume rise scheme is not intended to be used, and indeed cannot be used, with the dust or sea-salt schemes or with sedimenting species.



There is also an iterative plume-rise model within NAME for estimating source mass flux for volcanic eruptions, see Section 1.5.4.5. The iterative model does not aim to treat plume rise on a particle-by-particle basis, as the plume rise scheme does, but instead considers the bulk estimation of the emission mass flux based on the atmospheric conditions and observed rise height.

1.7.7 Deposition and sedimentation processes

Several mechanisms can remove material from the atmosphere to the ground surface. The processes of dry deposition, wet deposition and sedimentation (or gravitational settling) are represented in NAME. These loss processes can act in various different ways depending on the nature of the species and the characteristics of the underlying surface. For instance, different schemes might be applied for gaseous species, non-sedimenting particles and sedimenting particles.

1.7.7.1 Dry deposition

Dry deposition is the process by which material is removed from the atmosphere by transport to, and subsequent uptake by, the ground in the absence of precipitation. The uptake at the ground is strongly dependent on the land-surface type and the species.

Modelling of dry deposition is controlled in NAME via the variable [Dry Deposition?](#) in the [Sets of Dispersion Options](#) block of the main input file. This acts as a global switch for turning dry deposition on or off in a model run. Dry deposition in NAME is applied on a particle basis (that is, each particle can lose a certain proportion of its mass or activity to the underlying surface at each model time step) and these individual contributions are aggregated together into a deposition field.

The dry deposition parametrizations available in NAME are based on the concept of a *deposition velocity* v_d . The flux of pollutant to the ground, F , is proportional to the near-surface concentration of pollutant, C , and is given by

$$F = v_d C, \quad (1.7.6)$$

where the constant of proportionality v_d is given by the deposition velocity. The deposition velocity can either be a 'typical' value for that pollutant (which is specified by the user) or it can be determined indirectly using a resistance analogy,

$$v_d = \frac{1}{R_a + R_b + R_c}, \quad (1.7.7)$$

where R_a is the aerodynamic resistance, R_b is the laminar sublayer resistance and R_c is the surface resistance. All resistances are in units of sm^{-1} . The aerodynamic resistance represents the efficiency with which material is transported to the ground by turbulent motions in the atmosphere and is independent of the pollutant type. The laminar resistance describes the resistance to transport by molecular diffusion across the thin quasi-laminar layer adjacent to the surface and is formulated differently for gases and particles. In most circumstances R_b is smaller than R_a . The surface resistance characterises the resistance to capture by the surface and is dependent on both the pollutant and the underlying surface.

The above conceptual framework can be invoked in various forms with varying levels of complexity depending on the user's preference and nature of the species. It allows a distinction to be made between the treatment of gases and that of sedimenting or non-sedimenting particles. In broad terms, the modelling of dry deposition is governed by the four variables [Deposition Velocity](#), [Surface Resistance](#), [Mean aerosol diameter](#) and [Land use dependent dry dep](#) in the [Species](#) input block (see [DEFINING SPECIES INFORMATION IN NAME](#) in 1.5.1), where *no more than one keyword* can be specified for any given species.

Using an explicit deposition velocity

The most basic implementation of dry deposition in NAME is a simple parametrization scheme using the formulation (1.7.6) with a prescribed value of deposition velocity, v_d . This single value of v_d is assigned to a species using the [Deposition Velocity](#) variable in the species definition, and is used for all land-surface types (i.e. the scheme is independent of the underlying surface characteristics). The prescribed value of v_d must be non-zero for a sedimenting species.

The basic dry deposition scheme is applied whenever a deposition velocity is specified in the relevant species definition (and prescribing a deposition velocity in this way may potentially override other implementations discussed below). The basic scheme is universally applicable for all species (gases and particle types), although special treatment applies to two species, ozone and hydrogen¹³, which may mean that the user-specified deposition velocity is modified for these two species.

Using a resistance-based deposition velocity

When a deposition velocity is not given explicitly (i.e. the [Deposition Velocity](#) variable is blank), the dry deposition scheme used then depends on whether the species is modelled as a gas or as a particle. When modelled as a particle species, a resistance-based parametrization equivalent to (1.7.7) is applied, whereas three possible options are available for modelling the dry deposition of gaseous species: no dry deposition, a simple parametrization based on the resistance analogy (1.7.7) with a given fixed surface resistance R_c , or an advanced parametrization scheme based on (1.7.7) with a land-surface dependent surface resistance.

Consider first the treatment of non-gaseous pollutants which can be modelled as sedimenting or non-sedimenting particles. A sedimenting particle has a particle diameter¹⁴ and a density specified through the source term and is subject to gravitational settling. For such particles, any [Mean aerosol diameter](#) specified for a species is ignored.

A non-sedimenting particle (i.e., an atmospheric particle in suspension) has no particle diameter or density specified in the source term (in other words, no particle diameter or size distribution has been specified for the release) but uses a species for which the [Mean aerosol diameter](#) variable has been defined in the species definition. For sedimenting or non-sedimenting particles, the NAME parametrization of dry deposition is based on the resistance analogy (1.7.7) with the surface resistance R_c assumed to be zero. That is,

$$v_d = \frac{1}{R_a + R_b}.$$

The laminar sublayer resistance R_b is based on the work of Underwood [1999] and is dependent on the particle size. When this scheme is used, NAME takes the particle size to be the value of the [Mean aerosol diameter](#) assigned to the species for a non-sedimenting particle, or the particle diameter for a sedimenting particle. In the former case, the user needs to give an appropriate mean aerosol diameter to use (see [MD1/2](#) for further guidance). Note, however, that for sub-micron particles (below 1 μm in diameter) it is not necessary to know an accurate aerosol diameter since the laminar sublayer resistance parametrization does not depend on the diameter for particles of this size. As with the deposition velocity approach, this parametrization does not depend on the land-surface type.

¹³These rules apply different deposition velocities over sea areas and, in the case of hydrogen, a monthly variation in deposition velocities to reflect seasonal patterns (see code documentation for further details).

¹⁴Note that the particle diameter may vary from particle to particle when a particle size distribution is used.

For a gaseous species, the three options for alternative dry deposition schemes are invoked depending on the contents of the pair of variables [Surface Resistance](#) and [Land use dependent dry dep](#) in the [Species](#) block. At most, only one of this pair should be defined for each gaseous species (although different variables might be set for different species). If neither variable is given (recalling also that both the [Deposition Velocity](#) and [Mean aerosol diameter](#) variables are also blank), then there is no dry deposition for that species (this is equivalent to the user specifying a deposition velocity of zero). If a value is given for [Surface Resistance](#) then the resistance analogy (1.7.7) parametrization is applied using the prescribed value of R_c . As above, this single value for R_c is independent of the land-surface type. The final option is an advanced land-surface dependent scheme available for certain gaseous species, which is invoked by setting the keyword [Land use dependent dry dep](#) to [Yes](#).

A land-surface dependent dry deposition scheme exists for a selection of gaseous species (namely, ozone, nitric oxide, nitrogen dioxide, nitric acid, peroxyacetyl nitrate, hydrogen peroxide, methane, carbon monoxide, hydrogen, formaldehyde, sulphur dioxide and ammonia). It uses land-surface dependent surface resistance parametrizations from the Met Office global chemistry model, STOCHEM, to provide a detailed parametrization for the surface resistance term, R_c , representing the variety of deposition pathways that are available (although not all pathways are important for each species). For each gaseous species in the above list, an effective deposition velocity is then obtained by taking a weighted average of the individual deposition velocities for each land-surface type based on the fraction of that type in each model grid box. Variation of dry deposition rates between different land-surface types is evident. In particular, a strong land-sea effect is observed with ozone, nitrogen dioxide and peroxyacetyl nitrate.

The land-surface dependent dry deposition scheme requires additional surface and plant information (e.g. stomatal conductance, leaf area index, land use fractions). This additional information may be available as part of the standard NAME UM met files (for fields that vary on the same timescale as the meteorological fields) or as ancillary files (for data which are either constant with time or vary on a slower timescale than the meteorology such as monthly or seasonal variations). These additional fields are only available in the more recent UM met data sets (generally from 2009 onwards).

See [▷ MD1/2: A LAND-USE DEPENDENT DRY DEPOSITION SCHEME FOR NAME ◀◀](#) for further details, including detailed scientific documentation of the land-use dependent dry deposition scheme and aerosol scheme, further information for the model user on invoking these schemes, and an evaluation against the basic dry deposition scheme in NAME.

Specifying a maximum deposition height

Ordinarily, all model particles within the boundary layer at the end of each model time-step are subject to dry deposition in NAME. This is equivalent to assuming the boundary-layer average

concentration is representative of the near-surface reference concentration used in the formulation (1.7.6), and is an appropriate assumption if the pollutant is well-mixed within the boundary layer.¹⁵ However, in situations where pollutant is not well-mixed within the boundary layer (for example, when modelling near-source problems), this method does not provide an accurate representation of the dry deposition. To overcome this issue and allow more accurate modelling of dry deposition at short ranges, the concept of a deposition height, z_s , is introduced as a parameter in the model which defines the height below which particles are subject to dry deposition. The user can choose to control this value in situations, such as near-source problems, where pollutant cannot be assumed to be well-mixed within the boundary layer.

The deposition height (in metres) is set using the variable [Max deposition height](#) in the [Sets of Dispersion Options](#) block of the input file. The general recommendation for most types of NAME run is to leave the [Max deposition height](#) value blank, in which case the default is to use the boundary-layer depth. It may be appropriate to use smaller values of the deposition height for near-source studies, however the user might need to increase the number of model particles to overcome the increased computational noise, which will increase the cost of the model run.

Requesting deposition outputs

When dry deposition is requested as an output from NAME, the keyword [Sync?](#) should be set to [No](#) in the relevant request in the [Output Requirements - Fields](#) block. This is because deposition fields are calculated internally inside the particle loop by integrating the contributions at every advection time step. For radioactive species, which exhibit a half-life decay, the decay of deposited material over time is modelled by default, unless the variable [Decay deposition?](#) is set to [No](#) in the [Output Requirements - Fields](#) block. For further information on output requests, see [REQUESTING OUTPUT OF FIELDS FROM NAME](#) in 1.4.1.

1.7.7.2 Wet deposition

Wet deposition is the process by which material is removed from the atmosphere by transport to the ground in the presence of precipitation (rain, snow, etc.). For many pollutants it is the dominant removal process. Wet deposition comprises of two main processes: below-cloud or washout, where material is 'swept out' by falling precipitation elements, and in-cloud or rainout (which is often the more effective process), where material is absorbed directly into cloud droplets or ice crystals. Aerosols can act as cloud condensation nuclei in cloud droplet formation or can enter precipitation elements through impaction or interception. Wet deposition of gases depends on the solubility of the gas.

Modelling of wet deposition in NAME is controlled via the keyword variable [Wet Deposition?](#) in

¹⁵By averaging over the depth of the boundary layer, this approach also reduces statistical noise and provides for relatively smooth deposition fields.

the **Sets of Dispersion Options** block of the main input file. This acts as a global switch for turning wet deposition on or off.

Wet deposition is applied in NAME on a particle basis in a similar way to dry deposition. The removal of material from the atmosphere by wet deposition processes is based on the depletion equation

$$\frac{dc}{dt} = -\Lambda c, \quad (1.7.8)$$

where c is the air concentration, t is time and Λ is a scavenging coefficient. The mass of material represented on a NAME particle is depleted according to

$$\Delta m = m [1 - \exp(-\Lambda \Delta t)], \quad (1.7.9)$$

where m is the mass of the particle at the start of the timestep, and Δm is the change in mass over the timestep Δt . The scavenging coefficient Λ is defined by

$$\Lambda = A r^B, \quad (1.7.10)$$

where r is the precipitation rate in mm hr^{-1} , and A and B are parameters which depend on different types of precipitation (rain and snow) and on different deposition processes (rainout and washout).

Single-site precipitation data (for example, from observations) is input directly in units of mm hr^{-1} . Precipitation data from NWP models is read into NAME in units of $\text{kg m}^{-2} \text{s}^{-1}$ and converted internally to units of mm hr^{-1} . Precipitation data from NWP models is generally available as two components: dynamic (or large-scale) precipitation and convective precipitation. The former component being precipitation that is explicitly resolved by the NWP model (e.g. frontal rainfall), whereas the latter component represents smaller scale precipitation features (e.g. showers) that are parametrized in an average sense in the NWP model. For this reason, the area within any NWP gridbox is split into two parts: a fraction C_f , where C_f is the convective cloud fraction, which is within the region of convective precipitation, and the remaining fraction $(1 - C_f)$ outside of the region of convective precipitation. Within the region of convective precipitation, the convective precipitation rate, r_{con} is scaled by the convective cloud amount to give an effective convective precipitation rate, r_{con}/C_f . Hence the total precipitation rate within regions of convective precipitation is given by $r_{dyn} + r_{con}/C_f$, where r_{dyn} is the dynamic (large-scale) precipitation rate. An overall total scavenging coefficient is then used in equation (1.7.9) in NAME, viz.

$$\Lambda = (1 - C_f) A_1 r_{dyn}^{B_1} + C_f A_2 \left(r_{dyn} + \frac{r_{con}}{\max(C_f, 0.05)} \right)^{B_2}. \quad (1.7.11)$$

The two terms on the right hand side of equation 1.7.11 comprise relative proportions of scavenging coefficients in areas outside of convective showers and within convective showers, respectively. For precipitation data where separate large-scale and convective components are not available (for

example, when using single site observations or precipitation data from kilometre-scale resolution NWP models where convection is resolved explicitly), $C_f = 0$ and an overall total scavenging coefficient is given by equation 1.7.10 where the precipitation rate, r , is given by the total precipitation rate.

The A_1 , A_2 , B_1 and B_2 values are appropriate A and B scavenging parameters specified by the user in the **Species** block using the variables **A rain - BC**, **B rain - BC**, **A snow - BC**, **B snow - BC**, **A rain - IC**, **B rain - IC**, **A snow - IC** and **B snow - IC**, where **IC** refers to in-cloud (that is, rainout) values and **BC** refers to below-cloud (that is, washout) values. Negative values are not permitted for any of these variables (although zero values are allowed). Wet deposition can be turned off for a particular species by leaving all of the A and B scavenging parameters blank (this is the default). In addition, wet deposition by a particular process (e.g., washout by snow) can be turned off for a species by setting the appropriate A parameter to zero. Recommended values for A and B scavenging parameters (when precipitation rates are given in mm hr^{-1}) are listed in Table 1.25. These recommended values for the scavenging parameters are based on those used in the original NAME wet deposition scheme for aerosols designed by Dr T Choularton at UMIST and on other parameters given in the literature and used in other models. For gases which are insoluble or only slightly soluble, the scavenging parameters should be left blank.

The below-cloud values for A_1 , B_1 and A_2 , B_2 are used if the NAME particle lies below the relevant cloud base, whereas in-cloud values are used if the particle lies between cloud base and cloud top. Here dynamic cloud (when this is available as an input met field) might have a different cloud base and top to the total cloud (which includes the convective component also), in which case it is possible for a particle to use below-cloud A_1 , B_1 and in-cloud A_2 , B_2 values. Below-cloud, snow values are used if the temperature at the NAME particle position is less than 270 K and rain values are used otherwise. In-cloud, snow values are used if the temperature at the NAME particle position is less than 238.15 K, rain values are used if the temperature is greater than 273.15 K and, for temperatures in between, mixed-phase values are calculated by interpolating between the snow and rain values.

A critical value, PptCrit, of 0.03 mm / hr is implemented in the NAME wet deposition scheme to act as a filter for light drizzle. The reason for applying a threshold is that historically there has been a drizzle issue with the global version of the UM, which has tended to produce too much light precipitation. Implementing this threshold cut-off therefore ensures that this over-production of precipitation in the UM does not result in an over-prediction of wet deposition. If the dynamic precipitation rate is less than 0.03 mm / hr then the dynamic precipitation rate is set to zero. If the convective precipitation rate (before being scaled by the convective cloud amount to give an effective precipitation rate) is less than 0.03 mm / hr, then the convective cloud fraction and the convective precipitation rate are set to zero.

For further details on the NAME wet deposition scheme, see the Met Office Technical Report Webster and Thomson [2014].

Species	Rain mm hr ⁻¹		Snow/Ice mm hr ⁻¹ (water equivalent)	
	Below cloud	In cloud	Below cloud	In cloud
	A rain - BC B rain - BC	A rain - IC B rain - IC	A snow - BC B snow - BC	A snow - IC B snow - IC
Aerosols & Particulates	$A = 8.4 \times 10^{-5}$ $B = 0.79$	$A = 3.36 \times 10^{-4}$ $B = 0.79$	$A = 8.0 \times 10^{-5}$ $B = 0.305$	$A = 5.2 \times 10^{-5}$ $B = 0.79$
SO ₂	$A = 1.2 \times 10^{-4}$ $B = 0.79$	$A = 1.0 \times 10^{-4}$ $B = 0.79$	$A = 1.2 \times 10^{-4}$ $B = 0.305$	$A = 1.5 \times 10^{-5}$ $B = 0.79$
Ammonia	$A = 1.2 \times 10^{-4}$ $B = 0.79$	$A = 3.5 \times 10^{-4}$ $B = 0.79$	$A = 1.2 \times 10^{-4}$ $B = 0.305$	$A = 3.5 \times 10^{-4}$ $B = 0.79$
HNO ₃ & HONO	$A = 1.0 \times 10^{-4}$ $B = 0.79$	$A = 3.5 \times 10^{-4}$ $B = 0.79$	$A = 1.0 \times 10^{-4}$ $B = 0.305$	$A = 3.5 \times 10^{-4}$ $B = 0.79$
H ₂ S	$A = 2.0 \times 10^{-5}$ $B = 0.79$	$A = 1.7 \times 10^{-5}$ $B = 0.79$	$A = 2.0 \times 10^{-5}$ $B = 0.305$	$A = 2.6 \times 10^{-6}$ $B = 0.79$

Table 1.25: Suggested NAME scavenging parameters for different species

When wet deposition is requested as an output from NAME, the keyword [Sync?](#) should be set to [No](#) in the relevant request in the [Output Requirements - Fields](#) block. This is because deposition fields are calculated internally inside the particle loop by integrating the contributions at every advection time step. For radioactive species, which exhibit a half-life decay, the decay of deposited material over time is modelled by default, unless the variable [Decay deposition?](#) is set to [No](#) in the [Output Requirements - Fields](#) block. For further information on output requests, see [REQUESTING OUTPUT OF FIELDS FROM NAME](#) in 1.4.1.

1.7.7.3 Sedimentation

Sedimentation (or gravitational settling) of heavy particles are modelled in NAME using the concept of a sedimentation velocity. The gravitational settling scheme is automatically applied for any source that is set up to release sedimenting particles by defining a particle density [Particle Density](#) (in kg m⁻³) and either a particle diameter [Particle Diameter](#) (in μm) or a particle size distribution [Particle Size Distribution](#) (from which a diameter is calculated for each individual particle released) in the [Sources](#) block, see [SEDIMENTING RELEASES](#) in 1.5.4. A [Particle Shape](#) and associated [Sedimentation Scheme](#) can also be defined to model the sedimentation of non-spherical particles. Dust and sea-salt sources must be modelled as sedimenting particles but the gravitational settling scheme is also commonly used for other particulate sources such as volcanic ash.

NAME calculates a sedimentation velocity w_{sed} (in m s⁻¹) from the particle diameter, particle density, particle shape and ambient meteorological variables at the particle location [Maryon, 1997, Pruppacher and Klett, 1997]. The dynamic (or absolute) viscosity, μ , can be accurately estimated

from

$$\mu = \begin{cases} [1.718 + 0.0049 (T - 273.15)] \times 10^{-5}, & T > 273.15\text{K} \\ [1.718 + 0.0049 (T - 273.15) - 1.2 \times 10^{-5} (T - 273.15)^2] \times 10^{-5}, & T \leq 273.15\text{K} \end{cases} \quad (1.7.12)$$

where T is the ambient temperature in K at the particle position (see equations 10-141a and 10-141b on page 417 of Pruppacher and Klett [1997]).

The scheme used by NAME to calculate the terminal fall velocity of sedimenting particles in the atmosphere is outlined by Maryon [1997] and Maryon et al. [1999]. The terminal fall velocity w_{sed} of the particle with respect to the air is given by

$$w_{sed} = \left(\frac{4}{3} \frac{D}{C_D} g \frac{\rho_p - \rho_a}{\rho_a} \right)^{1/2}, \quad (1.7.13)$$

where D is the particle diameter (in m), ρ_p is the particle density and ρ_a is the ambient air density at the particle location. The drag coefficient, C_D , is, primarily, a function of the particle shape and the Reynolds number, Re , defined by

$$Re = \frac{\rho_a w_{sed} D}{\mu}. \quad (1.7.14)$$

Spherical particles

For spherical particles and $Re \ll 1$, the Stokes flow regime gives $C_D = 24/Re$ and

$$w_{sed} = \frac{D^2}{18\mu} g (\rho_p - \rho_a). \quad (1.7.15)$$

For spherical particles and $Re \gg 1$, the drag force C_D is a non-linear function of the Reynolds number, which can be well represented by a simple empirical equation [White, 1974]

$$C_D = C_1 + \frac{24}{Re} + \frac{C_2}{1 + \sqrt{Re}}, \quad (1.7.16)$$

where, for $Re < 5 \times 10^3$, $C_1 = 0.25$ and $C_2 = 6.0$. To solve for the terminal velocity of the particle (equation 1.7.13) requires an iterative approach in which the terminal velocity of a particle in Stokes flow is applied as an initial first guess.

Non-spherical particles

There are several schemes available in the literature which describe the drag coefficient of a *non-spherical particle*. In NAME, the Ganser [1993] and Wilson and Huang [1979] schemes are available to represent the fall velocity of non-spherical particles.

The semi-empirical drag coefficient defined by Ganser [1993] is given as

$$C_D = \frac{24}{Re K_1} \left(1 + 0.1118 [Re K_1 K_2]^{0.6567} \right) + 0.4305 K_2 \left[1 + \frac{3305}{Re K_1 K_2} \right]^{-1} \quad (1.7.17)$$

where

$$K_1 = \frac{3}{1 + 2\Psi^{-0.5}} \quad (1.7.18)$$

and

$$K_2 = 10^{1.8148(-\text{Log}\Psi)^{0.5743}} \quad (1.7.19)$$

The sphericity, $0 < \Psi \leq 1$, describes the degree to which a particle approaches a spherical shape, with the sphericity of a sphere being equal to 1. Sphericity can be defined as the ratio of the surface area of a sphere with equivalent volume to the actual surface area of the particle. Ganser [1993] provide sphericity values for a range of standard shapes, see Table 1.7.7.3. Sphericity can also be defined in 2-dimensions as the ratio between the projected area, A , to the square of the projected perimeter, P , as described by Riley et al. [2003]

$$\Psi_R = \frac{4\pi A}{P^2} \quad (1.7.20)$$

Alfano et al. [2011] suggest that the model of Ganser [1993] combined with sphericity, Ψ_R , as defined by Riley et al. [2003] is the best scheme for the calculation of the fall velocity of non-spherical volcanic ash particles, which have complex shapes (Figure 1.10).

Shape	Sphericity (Ψ)
Sphere	1.000
Cube Octahedron	0.906
Octahedron	0.846
Cube	0.806
Tetrahedron	0.670

Table 1.26: Particle sphericities Ψ given by Ganser [1993] for a range of standard shapes.

An empirical evaluation of drag for volcanic ash particles is given by Wilson and Huang [1979]

$$C_D = \begin{cases} \frac{24}{Re} F^{-0.828} + 2\sqrt{1.07 - F} & Re \leq 100, \\ \text{Linear Interpolation} & 100 < Re < 1000, \\ 1 & Re \geq 1000, \end{cases} \quad (1.7.21)$$

where F is a shape factor, $F = \frac{D_m + D_s}{2D_l}$ and D_l , D_m and D_s are the longest, intermediate and shortest axes of the particle.

When the size of the particle is of the same order of magnitude as the distance between gas molecules (i.e. submicron) the particle no longer moves as a continuum in the gas phase but instead as a particle amongst discrete gas molecules, which reduces the drag force acting on the particle. This is corrected for by multiplying by the Cunningham Correction Factor, CCF , defined as

$$CCF = 1 + \alpha N_{K_N} \quad (1.7.22)$$

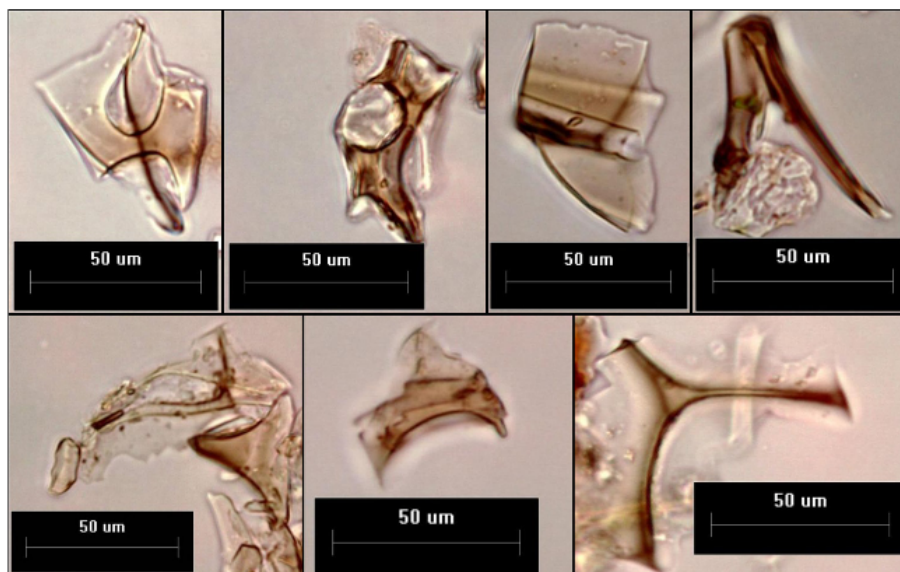


Figure 1.10: From Stevenson et al. [2013], tephra shards from the eruption of Grímsvötn in 2011 collected in rainwater sampled across the UK.

where

$$\alpha = 1.257 + 0.4 \exp \left(\frac{1.10}{N_{Kn}} \right) \quad (1.7.23)$$

and the Knudsen number is given by

$$N_{Kn} = \frac{2\lambda_a}{D}, \quad (1.7.24)$$

where λ_a is the mean free path of air particles given by

$$\lambda_a = \lambda_{a,0} \left(\frac{P_0}{P} \right) \left(\frac{T}{T_0} \right) \quad (1.7.25)$$

Standard conditions are given by Pruppacher and Klett [1997] to be $P_0 = 1013.25$ hPa, $T_0 = 293.15$ K and $\lambda_{a,0} = 6.6 \times 10^{-8}$ m.

Gravitational settling of particles contributes to dry deposition. The interactions between the gravitational settling scheme and the advection and dry deposition schemes are detailed in Webster and Thomson [2011]. The gravitational settling scheme puts additional restrictions on the internal advection time-step and hence may result in a smaller model time-step and consequently slower model run times. Turbulence parameters (namely, velocity variances and Lagrangian timescales) are modified for heavy particles to account for trajectory crossing and inertial effects (see [TURBULENCE SCHEMES USED IN NAME](#) in 1.7.4.3).

1.7.8 Radiological modelling

The radiological modelling capabilities of NAME include a basic scheme representing half-life decay of single radionuclides as well as more advanced options, such as the representation of decay

chains modelling disintegrations through a series of radionuclides, or the estimation of cloud gamma dose from airborne material ('cloud shine').

When modelling a radionuclide, material is typically represented in units of *activity* (e.g. the *becquerel Bq* or *terabecquerel TBq*) rather than the physical mass. In other words, the mass units are actually units of activity in this context and references to 'mass' and 'activity' tend to be used interchangeably in the following discussion.

1.7.8.1 Radioactive decay

For a radioactive species, NAME can model mass or activity loss due to radioactive decay processes. The governing equation involves an exponential decay

$$\frac{dm}{dt} = -\lambda m$$

where $\lambda = (\ln 2)/T_{1/2}$ is known as the *decay constant* (s^{-1}) and $T_{1/2}$ is the *half-life* of the species (in seconds). The species half-life is prescribed in the variable [Half Life](#) in the [Species](#) block of the input file (see [DEFINING SPECIES INFORMATION IN NAME](#) in 1.5.1). Note that for stable nuclides (that is, for species that are radiologically inert), the half-life value should be given as the text string [Stable](#), which ensures that no decay in mass or activity will occur for that species. The mass on a particle reduces over the time step t to $t + \Delta t$ as

$$m(t + \Delta t) = m(t) \exp(-\lambda \Delta t)$$

The above method is used whether the material is airborne on model particles or has been deposited on the ground.

Modelling of radiological decay is controlled using the global switch [Radioactive Decay?](#) in the [Sets of Dispersion Options](#) block, which should be set to [Yes](#) if the user requires decay to be considered. The user is able to select separately whether deposited radioactivity is subject to decay or not via the [Decay deposition?](#) keyword in the field request in the [Output Requirements - Fields](#) block. Specifically, when [Decay deposition?](#) is set to [No](#), NAME considers radioactive decay during the dispersion phase, but does not account for the radioactive decay of the deposited material; whereas setting the keyword to [Yes](#) instructs NAME to model both the radioactive decay during dispersion and the radioactive decay of deposited material for all radionuclides.



If the [Decay deposition?](#) keyword is not specified then the default in NAME is to include the radioactive decay of deposited material (i.e., equivalent to if [Decay deposition?](#) had been set to [Yes](#)).

1.7.8.2 Decay chains

In radiological problems it is often not sufficient to consider just a single half-life decay process, but instead becomes necessary to take into account the effects of radioactive decay chains. That is, a series of decay products arising from the decay of a parent radionuclide. NAME is able to model a decay chain for radioactive material which is dispersed in the atmosphere, but it does not currently treat a decay chain for material deposited on the ground.

In general, the serial transformation by radioactive decay of each member of a radioactive series is described by the Bateman equations (the derivation of which can be found in Fitzgerald et al. [1967]). Assuming that, at time zero, the radioactivity of the parent nuclide is A^0 and the radioactivity of every daughter is zero, then the radioactivity after a time t (in seconds) of the i -th chain member is

$$A_i(t) = A^0 \left(\prod_{j=1}^{i-1} f_{j,j+1} \lambda_{j+1} \right) \sum_{j=1}^i \frac{e^{-\lambda_j t}}{\prod_{\substack{k=1 \\ k \neq j}}^i (\lambda_k - \lambda_j)}$$

where $\{\lambda_j, j = 1, \dots, i\}$ is the collection of decay constants for the radioactive series, and $\{f_{j,j+1}, j = 1, \dots, i-1\}$ denotes the collection of branching ratios for the decay chain (that is, the fraction of nuclear transformations of each chain member that decay into the successive member in the chain).

NAME can calculate the radioactivity from a parent radionuclide and its daughter products for any *non-branching* decay chain or for a *single pathway* of a branching decay chain. In the latter case, NAME accounts for the branching ratio of each primary branch, but the radioactivity deriving from any secondary branches is assumed to be lost. For example, the radionuclide Sr-91 decays to two daughter products, Y-91 and Y-91M, and NAME can account for either one of these decay branches using the appropriate branching ratio but not both simultaneously (at least, not within the same decay chain instance). In other words, NAME could represent the decay pathway and associated branching ratio from Sr-91 to Y-91M but not the branching ratio to the other daughter Y-91, or vice-versa, and consequently the radioactivity decaying from Sr-91 to Y-91 would be lost from the assessment. If both pathways are significant, this issue can be resolved by defining two separate sources that must be distinguished using different species name (e.g. Sr-91a and Sr-91b) or, alternatively, two separate assessments can be performed.

Note that the secondary branch may not be significant, for example, if a parent predominantly decays to one daughter or if one daughter is much more radiologically significant. It should also be noted that if a radionuclide is released by a source but is also a daughter product of other radionuclides in the same release, then NAME will sum all contributions to give a single value for each radionuclide.

1.7.8.3 Cloud gamma dose

External exposure to gamma rays irradiating directly from a dispersing plume of radioactive material is referred to as *cloud gamma* and this may contribute significantly to dose if the radionuclides discharged are strong gamma emitters or are noble gases (which do not deposit, thus limiting the number of potential exposure pathways).

Two methods of calculating cloud gamma dose have been implemented in NAME: one scheme computes dose from each Lagrangian particle (and is referred to as the *Lagrangian particle* approach), and the other scheme computes dose from a gridded air concentration field (and is referred to as the *semi-infinite cloud* approach). The former scheme is more suited to inhomogeneous radioactivity concentrations in air at relatively short distances downwind, and the latter is more appropriate when modelling a large plume (relative to the distance over which gamma dose is significant) that is well-mixed within the boundary layer, typically at relatively large distances downwind.

The two cloud gamma dose schemes are implemented separately within NAME, and the user must decide which scheme to use or may consider using both in parallel. NAME does not include an automated capability for switching between the schemes or combining their results. A brief description of each scheme is presented below, see Bedwell et al. [2010] for further details.

The Lagrangian particle method

The Lagrangian particle approach (Raza and Avila [2001]) sums the contributions to dose at each receptor point from the radioactivity emitted by each individual model particle, assuming a point isotropic source. Specifically, the scheme is based on the following equation for the dose rate \dot{D}_γ (in Sv s⁻¹ or Gy s⁻¹) at the receptor point (x_0, y_0, z_0) as

$$\dot{D}_\gamma(x_0, y_0, z_0) = A k \sum_{p=1}^N \frac{f B(E_\gamma, \mu r) \exp(-\mu r) q_p(x', y', z')}{4\pi r^2}$$

where A is the effective or organ dose per unit air kerma (Sv Gy⁻¹ or Gy Gy⁻¹, respectively), k is the air kerma per unit fluence (Gy m²), p is the particle index (ranging from 1 to N , where N is the total number of model particles contributing to the dose for this radionuclide), f is the photon intensity, B is the build-up factor, E_γ is the photon energy released per disintegration (MeV Bq⁻¹), μ is the linear attenuation coefficient in air (m⁻¹), r is the distance between the particle and the receptor (m), and q_p is the radioactivity of the p -th particle (Bq), which is located at the point (x', y', z') . An upper bound is placed on the distance r such that insignificant doses are not calculated unnecessarily.

Default parameter values are adopted as follows:

- linear attenuation coefficient values as recommended by Hubbell [1982];
- Berger build-up factors as derived by Jaeger et al. [1968]; and
- air kerma per unit fluence and effective or organ dose per unit air kerma as given in ICRP

[1996].

Furthermore it is assumed that air is both the attenuation medium and the response medium (for photon flux calculations), that the irradiation at the point of the receptor is isotropic, and that all doses are calculated to adults only. However, alternative parameter values can be specified as input by the user if required and these will override default model values.

The semi-infinite cloud method

The semi-infinite cloud approach (Simmonds et al. [1995]) assumes that the radioactivity concentration in air is uniform over the volume of the plume from which photons can reach the receptor point at which the dose is to be calculated. It also assumes that the cloud is in radiative equilibrium, i.e. that the amount of energy absorbed by a given element of cloud is equal to that released by the same element. The following equation is used for the dose rate \dot{D}_γ (in Sv s⁻¹ or Gy s⁻¹) at the receptor point (x_0, y_0, z_0) :

$$\dot{D}_\gamma(x_0, y_0, z_0) = k C(x_0, y_0, z_0) \sum_{j=1}^n I_j E_j A_j$$

where $k = 2.0 \times 10^{-6}$ Gy y⁻¹ per MeV m⁻³ s⁻¹ is a conversion factor, $C(x_0, y_0, z_0)$ is the radioactivity concentration in air (Bq m⁻³) in the volume surrounding the receptor position, n is the total number of photon energies for any single radionuclide, I is the photon intensity, E is the photon energy (MeV), and A is the dose per unit air kerma (Sv Gy⁻¹ or Gy Gy⁻¹ depending on whether the dose is effective dose or organ dose per unit air kerma, respectively). This method uses grid-box averaged radioactivity concentrations in air that have been calculated from model particles.

The user is able to request the following cloud gamma dose outputs within an **Output Requirements - Fields** block:

Quantity	Description
<i>Adult Effective Cloud Gamma Dose</i>	Effective cloud gamma dose for an adult
<i>Adult Lung Cloud Gamma Dose</i>	Cloud gamma dose to the lung for an adult
<i>Adult Thyroid Cloud Gamma Dose</i>	Cloud gamma dose to the thyroid for an adult
<i>Adult Bone Surface Cloud Gamma Dose</i>	Cloud gamma dose to the bone for an adult

Additionally, the user needs to prescribe the method by which NAME is to calculate the cloud gamma dose for this field request (that is, whether to use the Lagrangian particle method or the semi-infinite cloud method). The calculation method is set using the logical keyword *Semi-infinite approx?* in the output field request. When set to **Yes**, the semi-infinite cloud approximation is adopted to estimate the cloud gamma dose using the large scale air concentration field. Otherwise the Lagrangian particle method is used to compute the cloud gamma dose at each receptor point

based on the radioactivity carried on the model particles.

NAME can estimate instantaneous and time-averaged effective and organ (thyroid, bone surface and lung) dose rates (in units of Sv s⁻¹ and Gy s⁻¹, respectively), and time-integrated effective and organ dose (in units of Sv and Gy, respectively). This is the same irrespective of which of the two calculation methods is applied. For scenarios involving multiple radionuclides, dose is presented as a function of radionuclide but it is not summed over all radionuclides.

When cloud gamma dose is being requested as an output from NAME, the user should ensure that an appropriate **Cloud Gamma Parameters** block is supplied for that radionuclide in the set of main input files. The input files may contain multiple (named) blocks of cloud gamma parameters, with each separate block identified by a unique name (normally these are labelled by the name of the radionuclide to which they relate). A cloud gamma parameters block is then associated with the relevant radionuclide using the keyword **Set of Cloud Gamma Parameters** in the **Species** block.



The files `Species.txt` and `Cloud_Gamma_Params.txt` in the **Resources** folder contain, respectively, all the radionuclide data and cloud gamma parameter data required for carrying out cloud gamma modelling in NAME. Sample NAME input files for runs where radioactive decay and cloud gamma dose are modelled are also included in the **Runs** folder.

1.7.9 Other decay processes

Other decay processes that are often non-radiological in nature can also be represented in NAME. These can decay mass or might completely remove (de-activate) material from the model simulation according to given criteria that are usually related to the meteorology in some way. In addition to a power law decay option and a simple generic scheme for representing decay of biological and chemical agents in sunlight, there are also several bespoke schemes currently implemented for various species. All these schemes are coded within the subroutine *AgentDecay* in `Particle.F90`. In principle, a user could include their own decay schemes in this routine based on the properties of the species under consideration.

The *agent decay* functionality includes a variety of decay schemes. A *power law decay* option is based on the formula

$$m(t) = m(t_0) \left(\frac{t_0}{t} \right)^k, \quad (t \geq t_0),$$

where the exponent k and the time delay t_0 before the power law decay starts are both positive and are specified in the **Species** input block (see **MISCELLANEOUS SPECIES CHARACTERISTICS** in 1.5.1.6). When used, power law decay is also applied to the decay of deposition fields (unless this is explicitly deactivated). Note that power law decay should not be used in combination with the standard radioactive decay based on species half-life.

A *UV decay* scheme is intended to represent the mass loss of a biological or chemical agent

due to the action of ultra-violet radiation in sunlight. The UV decay is represented using a nominal loss rate (expressed as a percentage loss per hour, between 0% and 100%), which is defined as part of the species information using the [UV Loss Rate](#) keyword (see [DEFINING SPECIES INFORMATION IN NAME](#) in 1.5.1). The nominal loss rate is defined based on the assumption of an overhead sun (i.e., a solar zenith angle of zero degrees) and this theoretical maximum value is then modulated by the actual zenith angle at any specific time and location. However no corrections are applied for any cloud cover (i.e., the scheme assumes clear-sky conditions).

The additional bespoke schemes currently included in NAME are:

Species: FOOT-AND-MOUTH / TCID50 deactivation of viable foot-and-mouth disease virus by low relative humidity below 50% (it assumes a virus loss rate of 50% per hour in dry air).

Species: MIDGE removal of midges by moderate or heavy rainfall in excess of 1 mm/hr.

Species: HYDROGEN removal of hydrogen via oxidation by hydroxyl radical OH (based on monthly climatology of OH concentrations).

Modelling of these other (mostly non-radiological) decay processes is invoked by setting the keyword [Agent Decay?](#) to [Yes](#) in the [Sets of Dispersion Options](#) block of the main input file.

1.7.10 NAME chemistry modelling

The inclusion of chemistry in the dispersion modelling framework opens up an extended range of functionality for NAME, including air quality forecasting and pollution episode analysis. However users are advised to seek further guidance from us before attempting to use the chemistry modelling options in NAME, because some expertise is required in setting up the model to produce meaningful results.

The model code is structured so that NAME can be run with different chemical schemes. This flexibility is achieved at a coding level by separating the chemistry scheme itself from its interaction with the rest of the NAME model. Here the chemistry scheme is selected via the module supplied in [ChemistryScheme.F90](#), while the interface with the wider code is provided by the [Chemistry.F90](#) module. In this way, alternative chemical schemes can be invoked by having different versions of the [ChemistryScheme.F90](#) file. Currently two schemes are available: a comprehensive gas and aqueous phase, tropospheric chemistry scheme and a simple scheme to allow conversion of gaseous isotopes of iodine to particulate iodine. The tropospheric chemistry scheme is stored in [ChemistryScheme.F90](#) (and is the default scheme that is used) and the iodine scheme is stored in [ChemistrySchemeIodine.F90](#). The user must take care to rename which ever scheme they require as [ChemistryScheme.F90](#) before compiling the code. In both cases the chemistry scheme is invoked by setting the input variable [Chemistry?](#) to [Yes](#) in the [Sets of Dispersion Options](#) block of the main input file.

1.7.10.1 Tropospheric chemistry scheme

When invoking the tropospheric chemistry scheme, the user needs to provide various supplementary information defining the appropriate species to be used with the NAME chemistry scheme and a suitable collection of grids for defining the chemistry fields (as Eulerian fields). Specifically, use of the chemistry scheme requires a specific collection of species to be defined (including details on whether these species are to be modelled on particles or as fields). Sample species files for use with the NAME chemistry scheme are provided in the resources folder [/Resources/Defns](#) to assist users here. Furthermore, horizontal and vertical grids need to be established defining both the (user-defined) three-dimensional Eulerian chemistry grid to be used in the NAME simulation and the (fixed) grid for the STOCHEM monthly-average background fields that are read into NAME. These grids have 'hard-wired' names, viz.

[Eulerian HGrid](#)

[Eulerian ZGrid](#)

[Eulerian ZGridBoundary](#)

for the run-specific Eulerian grid used for the chemistry fields (and set by the user), and

[STOCHEM - horizontal grid](#)

[STOCHEM - vertical grid](#)

[STOCHEM - interface levels](#)

for use with the fixed background fields generated by the STOCHEM global chemistry model. Note that each pair of horizontal and vertical grids should be interpreted as defining the centres of the relevant three-dimensional chemistry gridboxes, whereas the 'boundaries' levels or 'interface' levels refer to the bottom and top of each slab. The vertical grids for the main chemistry fields must use a z-based eta coord system if advecting Eulerian fields. The STOCHEM grid definition and an example of a NAME chemistry grid definition are again provided in the [/Resources/Defns](#) folder.

Further control of the chemistry scheme is provided by the **Chemistry Options** block in the main input file. The user has the ability to define a different folder to the default one for the storage of data files supplying NAME with monthly-average background concentration fields. By default, NAME will read this data from the folder [/Resources/Stochem](#) but this location can be changed via the keyword variable [Chemistry Folder](#).

The chemistry scheme was developed to utilise meteorological input fields from the Unified Model, although it has also been run with re-analysis datasets from ECMWF. Conceptually, meteorological fields from any NWP modelling system could be used provided that the relevant parameters required by the chemistry scheme are available.

Chemistry calculations are performed on a fixed three-dimensional chemistry grid at each model synchronisation time. For species held on particles, the initial species concentrations in a chemistry gridbox are first obtained by summing the contributions from all particles occupying that box at the given time. Following completion of the chemistry calculations, the updated mass of each species in

the chemistry box is then reassigned back to these particles: primary pollutants being redistributed according to the relative proportion of the original contributions, and with secondary species being distributed among particles carrying the appropriate primary species in proportion with the original amount of primary pollutants.

The initialisation of chemistry background fields at the start of a chemistry run is controlled using the **Category** value in the **Species** block. The category must be set to **CHEMISTRY-FIELD-INIT** if the field is to be initialised from a field read from a (STOCHEM) file and to **CHEMISTRY-FIELD** if the field is to be initialised to zero. Currently only H₂O₂ and O₃ can be initialised from an external file in this way. Note that species held on particles have the category **CHEMISTRY-SPECIES**, although the category has no effect for species on particles except that it is reproduced in the output.

When running NAME with chemistry, the synchronisation time interval is normally set to 15 minutes (that is, **Sync Time** is **00:15**). Chemistry fields and particle masses are therefore updated every 15 minutes. Output fields for species modified in the chemistry scheme should be requested as 'sync' output (that is, where **Sync?** is set to **Yes**) so as to ensure that output is calculated after the chemistry update step. In principle, it would be possible to run NAME with chemistry using a different synchronisation step. The chemistry mechanism has an internal time-step of 100 seconds, and the main synchronisation step is constrained to be a multiple of this value. However use of a sync time other than 15 minutes, although possible, would require careful consideration and testing. Although there are constraints on the sync time, the near-source scheme can still be applied with chemistry runs as it will automatically determine a shorter internal time step, where necessary, for advecting particles.

The NAME chemistry scheme was originally developed to model sulphate aerosol (Malcolm et al. [2000]) produced from primary emissions of sulphur dioxide (SO₂), nitric oxide (NO) and ammonia (NH₃). The scheme was subsequently revised to improve the parametrization of winter-time sulphate aerosol (Derwent and Malcolm [2000]) and to include nitrate aerosol chemistry (Redington and Derwent [2002]). The current chemistry scheme has undergone further development to include the release of primary emissions of carbon monoxide (CO), formaldehyde (HCHO), ethylene (C₂H₄), propylene (C₃H₆), isoprene (C₅H₈), o-xylene (C₆H₄(CH₃)₂), toluene (C₆H₅CH₃) and 1,3-butadiene (C₄H₆). The emissions from the seven volatile organic compounds (VOCs) are then scaled within the model in order to represent the released mass of the full VOC emission inventory.

The free radicals hydroxyl (OH) and hydroperoxy (HO₂) are modelled explicitly within NAME and ozone (O₃) and hydrogen peroxide (H₂O₂) are also calculated within the model, being initialised at the start of a model run using values taken from background fields provided by the Met Office global chemistry model, STOCHEM (Collins et al. [1997]). The user can make a choice by editing the chemistry species definition file as to which photo-oxidants are modelled on background particles, and which remain static on a three-dimensional field. Modelling long-lived species such as O₃ and H₂O₂ on model particles so that they can be advected throughout the domain is the preferred set up, however this necessitates considerable numbers of particles throughout the chosen model domain

to avoid statistical noise, and also an influx of the photo-oxidants at the domain boundaries. For short-lived species such as OH and HO₂, modelling on a fixed three-dimensional grid is considered to be acceptable. For model runs that either cover a very large domain, or need to be run fast, it may be necessary to model species such as O₃ and H₂O₂ on a fixed grid. That is, they will not be advected by the three-dimensional wind field unlike the species carried on model particles. Hence concentrations vary with the local chemistry, but the effects are not spread downwind. This is a good approximation for short lived species such as OH and HO₂, but is less suitable for species with longer atmospheric lifetimes such as H₂O₂ and O₃. Comparison of hourly ozone values at UK measurement sites has shown significantly better results using the approach where ozone is carried on background particles. Work is planned to allow advection of chemistry fields in the Eulerian framework, which should overcome this problem.

Redington et al. [2001] documents all the chemical reactions in NAME and the reaction rates and coefficients can be found in Collins et al. [1997] with a few exceptions. The ammonia and nitric acid equilibria with ammonium nitrate is based on the scheme presented in Ackermann et al. [1995]. Model ammonia is rapidly combined with available aerosol sulphate to form ammonium sulphate. Any ammonia that remains will form a thermodynamic equilibrium with nitric acid and ammonium nitrate (Meng and Seinfeld [1996]) which is dependent on temperature and relative humidity. The formation of coarse mode aerosol nitrate is parameterised by the reaction of N₂O₅ and nitric acid with natural dusts and sea salt and is described using a first order rate coefficient following the approach used by Abdalmogith and Harrison [2005]. It is well known (Dentener and Crutzen [1993]) that scavenging of N₂O₅ by water may be an important source of nitric acid. The reaction is not thought to be very fast as a homogeneous process but should occur as a droplet process. This has been included as a first-order conversion process with a temperature dependency to ensure that the process occurs less in summer months when less cloud is available. In principle a more sophisticated process could be put in place utilising the NWP cloud liquid water amount. It is also well known (Harrison and Kitto [1994]) that there are heterogeneous sources of HONO. A simple scheme has been put in place to effectively convert NO₂ to HONO during winter-time high NO_x conditions.

Sulphate aerosol is produced in the gas phase by reaction of sulphur dioxide with OH to form SO₃ which is then instantaneously converted into particulate sulphate. The aqueous phase oxidation of sulphur dioxide by hydrogen peroxide and ozone also makes an important contribution to the formation of sulphate aerosol. Aqueous phase chemistry is carried out if the NWP cloud fraction and cloud water are non-zero in the grid box. At the end of the time step it is assumed that the cloud has evaporated leaving particulate aerosol, but this would be automatically dissolved again at the start of the next chemistry time step if the cloud were still present. The reactions and equilibrium included in the NAME scheme are given in Redington et al. [2001]. The aqueous phase mechanism used in the NAME model is based on that used in STOCHEM (Collins et al. [1997]). The gaseous species dissolved into the cloud are nitric acid (HNO₃), ozone (O₃), carbon dioxide

(CO₂), hydrogen peroxide (H₂O₂), ammonia (NH₃) and sulphur dioxide (SO₂). The gas-to-liquid phase equilibria for the incorporation of soluble species into cloud droplets is described by Henry's law. Dissolved species such as SO₂ dissociate, which means that these species are more soluble than Henry's law suggests. The hydrogen ion concentration [H⁺] and the pH are calculated from the equation of acid-base balance using a minimisation approach. Sulphate aerosol in the cloud is assumed to be completely dissolved. Any ammonium present is assumed to react instantaneously with the sulphate to form ammonium sulphate. The loss of dissolved, gaseous species in the aqueous phase are passed through to the gas phase calculations, for example hydrogen peroxide can be rapidly completely depleted in the aqueous phase.

The NAME chemistry scheme also includes a simple scheme to produce both anthropogenic and biogenic secondary organic aerosol (SOA). The first step is to form a secondary biogenic organic aerosol precursor from α -pinene (C₁₀H₁₆) followed by a simple rate coefficient to model the subsequent gas to particle conversion of the precursor to form SOA. Loss of the precursor also occurs via oxidation by OH. Anthropogenic secondary organic aerosol (ASOA) is thought to be formed from atmospheric oxidation of a wide range of man made VOCs. Derwent et al. [2010] recently reported a study that developed the concept of secondary organic aerosol potential (SOAP) that reflects the propensity of each organic compound to form SOA. This was combined with a highly speciated emissions inventory to give the top 15 most prolific man-made SOA forming organic compounds over North-West Europe. Toluene was found to be the most prolific source of ASOA, and hence is used (scaled to represent the total of VOCs that form ASOA) in a simple scheme to form ASOA in NAME.

1.7.10.2 Iodine chemistry scheme

The alternative chemistry scheme available in `ChemistrySchemeIodine.F90` can be used, with radioactive decay, to model the conversion of gaseous radioactive iodine to particulate radioactive iodine. The species file is set up as detailed earlier, except that the *Category* is *RADIONUCLIDE* rather than *CHEMISTRY-SPECIES*. A suitable chemistry grid must be defined. In the *Sets of Dispersion Options* block, the input variables *Chemistry?* and *Radioactive Decay?* should both be set to *Yes*. To compile the code for Iodine chemistry, the file `ChemistrySchemeIodine.F90` should be renamed as `ChemistryScheme.F90` (after renaming the standard tropospheric scheme to a suitable file).

The iodine emissions can be in terms of mass or Bq, however certain parts of the radioactive decay scheme do not work for mass (e.g., calculation of cloud gamma dose). If an emission is in Bq the code converts it to mass for the chemistry. Note that *Molecular Weight* must be defined in the *Species* input block for the radioactive species. The *Material Unit* refers to the unit in which the material is to be emitted and can therefore be *g* or *Bq* in this case. Additional radioactive species required by the user may also be modelled. These are listed in the species file as normal and will be ignored by the chemistry scheme.

1.7.11 Eulerian model

An Eulerian model has been developed within NAME to represent the evolution of a scalar field rather than following the motion of individual Lagrangian particles. As such, the Eulerian model gives the concentration of a particular species directly rather than by averaging over the total mass on all the particles in an arbitrary grid box volume. The model uses a semi-Lagrangian method to solve the advective part of the scalar transport equation. The functionality of the Eulerian model includes vertical diffusion, sedimentation of sedimenting material, and dry and wet deposition. It is a hybrid model in that particles are initially released from a source as usual and the material associated with each particle is then transferred to the scalar field after a user-specified time defined separately for each source. This travel time for the transition onto the field can be specified as zero, in which case the material from that source is put onto the field immediately. The model can be used either as a global model or in a limited-area model configuration. The Eulerian model is integrated with NAME's chemistry scheme and is well suited to modelling the dispersion of species such as ozone that evolve smoothly over large spatial scales. The main features of the model are described below; see [MD1/4: EULERIAN-LAGRANGIAN HYBRID MODEL](#) for further information on the Eulerian model within NAME.

The Eulerian model is activated on a per-species basis by setting the [On Fields?](#) entry to [Yes](#) for the relevant species in the [Species Uses](#) block, see [DEFINING SPECIES USE INFORMATION IN NAME](#) in 1.5.2. A species can be solely on a field, for instance an initialised or evolved chemistry field, in which case the [On Particles?](#) entry would be [No](#). Whereas for primary pollutants released on particles and subsequently transferred to the field, both [On Particles?](#) and [On Fields?](#) should be set to [Yes](#). Then the travel time at which the mass held on particles from a given source is transferred onto the Eulerian field is a user-defined quantity, [Lagrangian-Eulerian Time](#), which is specified in the [Sources](#) block. This transition time can vary from one source to another, and if not specified takes a default value of infinity, which means that no mass will be transferred to the field for a given source unless a finite transition time is specified. Species for which the [On Fields?](#) entry is [No](#) do not respond to [Lagrangian-Eulerian Time](#) and remain on particles throughout. In the general case, therefore, there may be a mixture of species that are always retained on particles and some that are moved onto the Eulerian field. Finally, it is also necessary to specify a computational grid for the Eulerian model using some recognised 'hard-wired' names, as for the chemistry scheme (see page 128).

1.7.11.1 Semi-Lagrangian scheme

The semi-Lagrangian scheme used by the Eulerian model in NAME is based on the scheme used in the *ENDGame* dynamical core of the Unified Model. The method calculates the departure point of a fictitious particle arriving at a grid point of the Eulerian grid (the combination of Eulerian grid and Lagrangian particle gives the method its name). The values of the scalar field at the grid point and

at the departure point are then used to calculate the change in the scalar concentration (and hence its advection). The method is computationally stable even for large time steps. Further advantages of this method include monotonicity and positivity (i.e., sign preservation) so that spurious maxima or minima or changes of sign cannot be introduced in the advection of the scalar field. However, disadvantages of the method include potentially significant numerical diffusion which can be partly mitigated by using high-order interpolation schemes. For more information on semi-Lagrangian methods, see, e.g., Staniforth and Côté [1991].

Note that it is assumed there is zero concentration of each scalar field on the boundaries when using a limited area set up of the Eulerian model.

1.7.11.2 Vertical diffusion

The model uses a Crank-Nicolson method to calculate the vertical diffusion of the scalar field. This method is a finite difference method that is typically used for solving (numerically) a parabolic partial differential equation (pde) such as the heat (or diffusion) equation. It is second-order convergent in time so that the local truncation error associated with the time step is the same order as the spatial step. In practice, this means that the time step and spatial step can be the same order of magnitude (in some non-dimensional sense). The Crank-Nicolson method uses the average value of the spatial derivative at the beginning and end of the time step. This makes it unconditionally stable, i.e., errors are damped rather than magnified. It is an implicit method which requires the solution of a system of algebraic equations, which is achieved by means of Crout factorisation. The method can also be applied to a spatially irregular grid.

The lower boundary condition is a flux of material to the ground with a species and size-dependent (for sedimenting material) deposition velocity (see below for more details). If dry deposition is not activated then the deposition velocity is zero. The upper boundary condition requires zero flux of the scalar for all times.

Vertical diffusion is invoked by setting the [Turbulence](#) flag to [Yes](#) in the [Sets of Dispersion Options](#) block.

1.7.11.3 Sedimentation

Sedimentation occurs automatically in the Eulerian model after material has been transferred from sedimenting particles, i.e., those particles that have a defined particle size and density and are therefore subject to gravitational settling (here particles could have a range of different sizes if the release has been specified using a particle size distribution rather than a fixed diameter). In the Lagrangian model, the sedimentation velocity is calculated separately for each particle (since it is size dependent). For a field representing particles with a range of different sizes this is clearly not possible. In the Eulerian model, the sedimentation is calculated using the geometric mean of each size range bin in the particle size distribution. The calculation makes use of the same formulation

for the terminal velocity as is used for Lagrangian particles. The modified vertical velocity field is then passed to the semi-Lagrangian advection scheme.

1.7.11.4 Dry deposition

When modelling dry deposition, a deposition velocity is calculated at the lowest vertical level in the Eulerian field using the same function as is used for Lagrangian particles. For sedimenting particles, the representative diameter of the particles (which is taken to be the geometric mean of the relevant particle size bin) is used for this calculation.

Dry deposition is invoked by setting the [Dry Deposition](#) flag to [Yes](#) in the [Sets of Dispersion Options](#) block.

1.7.11.5 Wet deposition

When modelling wet deposition, the scalar field decays exponentially according to the formula

$$\chi = \chi \exp(-\Lambda \Delta t)$$

where χ is the scalar concentration and Δt is the time step. The scavenging coefficient, Λ , is dependent on both the species and the precipitation and is calculated using the same approach as used for Lagrangian particles.

Wet deposition is invoked by setting the [Wet Deposition](#) flag to [Yes](#) in the [Sets of Dispersion Options](#) block.

1.7.12 Short-term concentration fluctuations

Short-duration fluctuations in concentration are created within a dispersing plume by the action of turbulent diffusion processes, which gradually mix clean ambient air into the plume material. These processes are eventually responsible for mixing the plume with its surroundings at the very small dissipation scales where molecular diffusion can ultimately smooth out any fluctuations. The effects of fluctuations (including the magnitude of short-term peaks in concentration) therefore tend to be greatest at short ranges downwind from a compact source, with greater opportunity for dilution of contaminant material occurring as a plume travels further downwind.

Concentration fluctuations can be important in a wide variety of atmospheric dispersion problems. For instance, in assessing the ignition risk for flammable and explosive materials, in modelling the physiological response to some toxic compounds and in influencing chemical reactions where reaction rates are non-linear functions of the constituent concentrations. Fluctuations can also play a role in the design of source detection strategies, the perception of malodour and similar environmental nuisance issues, and the consideration of plume visibility for obscurant materials.

NAME incorporates a primitive fluctuations scheme designed to estimate certain statistics of concentration fluctuations for a simple point-source release scenario. The fluctuations scheme is automatically invoked to meet any requests for a standard deviation field or a probability distribution of short-term concentration, see [REQUESTING OUTPUT OF PDFS FOR CONCENTRATION FLUCTUATIONS](#) in 1.4.3. The NAME fluctuations scheme is limited, at present, to the prediction of fluctuation statistics for instantaneous concentrations.

The fluctuations scheme in NAME is designed to predict two aspects of concentration fluctuations experienced downwind of a *single continuous point source* within the atmospheric boundary layer:

- an estimate of the standard deviation of concentration, $\sigma_c = \sigma_c(x, y, z, t)$
- an estimate of the probability distribution of concentration values, expressed either as a collection of probabilities for exceeding specified concentration thresholds $\{c_\alpha\}$, viz. $p_{c_\alpha} = p_{c_\alpha}(x, y, z, t) = \text{Prob}\{c(x, y, z, t) \geq c_\alpha\}$, or as a collection of percentiles of concentration, $c_{p_\beta} = c_{p_\beta}(x, y, z, t)$, where $\text{Prob}\{c(x, y, z, t) \leq c_{p_\beta}\} = p_\beta$.

The standard deviation σ_c of the concentration fluctuations is processed as a field in NAME. This field can be requested as an output in its own right (by setting up a standard field request with the [Quantity](#) keyword given as [Sigma C](#)), or will otherwise be calculated automatically as an internal field whenever a corresponding request for a probability distribution is supplied. In fact, calculation of a standard deviation field depends, in turn, on several extra fields being available on the same grids: a mean concentration field, an average travel time field and X-Stats information (particle displacement statistics as a function of their travel time). These additional fields are prerequisites in the standard deviation calculation, and their associated field requests are again set up automatically, as and when required, to ensure that the relevant internal fields are available (if they have not already been requested explicitly by the user). The fluctuations scheme calculates an estimate of σ_c separately at each individual grid-point for each time step; further details of the calculation are described in [MD6/1: A FLUCTUATIONS SCHEME IN NAME III](#).

The calculation of a probability distribution for concentration fluctuations uses the mean concentration \bar{c} and standard deviation of concentration σ_c fields on the same spatial and temporal grids. Then, at each individual grid-point location and for each time, the fluctuations scheme calculates parameters of a *clipped Gaussian distribution* matching these predicted values of \bar{c} and σ_c . The exceedence probabilities or percentiles are then derived analytically from the calculated distribution.

The fluctuations scheme in NAME is based on the scheme developed by David Thomson for use in the ADMS modelling system (see ADMS documentation Thomson [1992, 1996, 2000]), although there are some significant differences between the two implementations of the scheme that essentially arise out of the contrast between the computational approaches of ADMS and NAME. In particular, some inputs are calculated differently (e.g., NAME obtains estimates of the plume spread

parameters σ_y and σ_z calculated directly from particle trajectories (X-Stats information) rather than via the mean concentration field).



The modelling of concentration fluctuations in NAME is currently intended as a research option only. Its functionality is limited and has not been tested as extensively as other, more mature, components of the model. The user should be aware that the fluctuations scheme is currently very restrictive in terms of the scenarios that can be represented. Specifically, it is limited to the consideration of instantaneous concentrations (i.e., with no time averaging) in a plume from a single, continuous, point source emission. Any user interested in exploring the fluctuations scheme should contact us to seek further advice.

1.8 Code Management, Compilation and Optimisation

This section gives a brief introduction to some of the more technical aspects of NAME surrounding code management and execution. The code structure and development practices are mentioned briefly in the first part below, although these topics will be discussed in much greater detail in the [NAME DEVELOPER GUIDE](#) (in preparation). Compilation of the NAME source code to produce runtime executables is then discussed, including the various compilation options that are available to the user. Methods of running NAME are considered – NAME can be run from the command line or launched via a script or batch file. The section concludes by examining the ability to run a parallelised version of NAME by using multiple threads on a shared memory multi-processor system. Other optimisation techniques will also be discussed, including use of timers within NAME.

1.8.1 Code structure and programming standards

NAME is structured into a collection of folders comprising of source code, pre-built executables, definitions and other resources, example runs, and various documentation. Other utilities are also provided for use with NAME, some of which are external packages that have been developed outside of the Met Office. A full description of the NAME distribution is given within the [INSTALLATION AND TROUBLESHOOTING GUIDE](#).

The NAME model is under continual development to improve and enhance its features and capabilities. A small team of model developers is active within the Met Office and other code improvements are fed back to us by external users. Indeed, feedback of external code developments is encouraged as these improvements can then benefit the entire NAME modelling community in the future. NAME code is managed internally under FCM, a software package providing a version-control environment based on *subversion* alongside a flexible issue-tracking capability.

See the separate [NAME DEVELOPER GUIDE](#) for further information on the structure of the NAME package and for guidance on code development practices and programming standards.

1.8.2 Compiling NAME

The NAME source code is written in the Fortran 90/95 programming language, with some use of compiler directives to control the compilation. A small amount of code preprocessing is also performed using a bespoke preprocessor program.

Let us start with a few of the basics (experienced programmers can skip to the next paragraph!). Computer programs are written in a human-readable form known as *source code*. The source code contains the set of instructions that the computer needs to follow in order to perform the intended task of that program. However a computer cannot interpret the source code directly and it therefore needs to be translated or *compiled* into a machine-readable form. This is the task of a Fortran compiler, which generates files known as *object code*. The object files need to be combined to

produce an *executable* file, which can then be run as a standard program on a computer. Combining the object files is the task of the linker (sometimes referred to as a link editor).

The NAME distribution supplied to external users already contains some pre-built executables for supported operating systems (RedHat Linux and Windows). These were compiled by the Met Office using the Intel Fortran compiler and should provide a usable selection of run-time executables for most user applications. Using these pre-built executables allows NAME to be run straight 'out-of-the-box' without needing to go to the effort of compiling from source code. This is especially useful if a Fortran compiler is not readily available on the computer that NAME is being installed on.

On the other hand, however, there are also some disadvantages in not recompiling source code when installing NAME on a new system. The supplied executables might not be fully optimised for the new configuration and recompilation could potentially create a faster executable program. There are also a few occasions when local library files are not installed or are incompatible with our build of NAME (this is especially the case when using OpenMP parallelisation or add-on functionality such as the ECMWF GRIB API). Recompiling is often the best method of resolving such difficulties.

The ability to compile NAME source code from scratch becomes essential, of course, if a user is intending to edit the code in any way. This could involve simple edits, such as modifying global parameters for their specific application, or more advanced code modifications, e.g., for research purposes. Advanced users of NAME will need to ensure that a suitable Fortran 90 compiler is installed on their system.

The precise details of compiling NAME will depend on the Fortran compiler (and the operating system) being used. General guidance on compilation with the **Intel Fortran compiler** is given here for the supported operating systems (RedHat Linux and Windows). However NAME is written in standard Fortran 90 (with a very few exceptions) and should therefore compile under any operating system provided that a suitable Fortran 90 compiler is available. For instance, NAME has been successfully compiled under Sun Solaris and Apple Mac O/S.

When compiling NAME on Linux systems, one of two compilation methods is generally adopted: use of a *compilation script* or use of a *makefile*. A compilation script provides a straightforward and relatively basic approach to the compilation in which the Fortran compiler is invoked using relevant options and commands to build an executable from the sources files in a static way. All source files are compiled from scratch, even if valid object code already exists for one or more of these files. Three pre-defined compilation scripts are supplied in the `Code_NameIII` folder to generate (serial) 'release' and 'debug' versions of NAME, as well as a 'release' version supporting use of the ECMWF GRIB-API package. Each of these scripts can be called via the command line in the usual way.

A 'makefile' has also been developed for NAME and provides an alternative method to using one of these compilation scripts. MAKE is a standard Linux utility that automatically determines which components of a program need to be compiled (or recompiled) based on given dependencies and relationships between files, and then issues the various compile and build commands to generate an executable. The 'makefile' describes these relationships amongst files in the program and supplies

instructions for updating each file (the executable file is updated from the object files, which are in turn built by compiling the source files). NAME uses the GNU implementation of MAKE. The advantage of using a 'makefile' over a basic compilation script is that only those components of the program that require updating are actually processed by MAKE. If an object file already exists and is up to date relative to its source file, then that compilation step can be skipped over. As a result, code compilation using a 'makefile' is often much faster than a full recompilation of all source files (especially, e.g., when making small changes to source files during code development activities).

The MAKE utility needs to be invoked using the supplied makefile in the `Code_NameIII` folder. Typing `make` (with no arguments) builds an executable based on the default compilation options (this creates a parallelised 'release' version of NAME) but other compilations options can be selected as additional arguments in the call to `make`; see comments in the 'makefile' for further details. The command `make clean` is also useful in removing all compiled objects and therefore forces a full compilation from scratch (this is necessary if the compilations options are modified in any way).



The MAKE utility is also supported on other (Unix-style) operating systems and the makefile supplied with NAME could be used for these other environments (possibly after some modification).

Compilation under Windows XP and Windows 7 uses the Intel Visual Fortran Studio. A *solution file* `NameIII.sln` is available in the top-level folder of the NAME distribution, which is designed to load the NAME project and its related projects into the Visual Fortran Studio tool. Here the related projects include the preprocessor utility (required to build some of the Fortran source files during the NAME compilation process) and the NAME2ADMS file conversion utility. Individual 'projects' are saved as `.vfproj` files in the relevant code folder of the NAME distribution. After loading these projects, the Intel Visual Fortran Studio provides a convenient graphically oriented user environment for code development, compilation and debugging purposes.

1.8.2.1 Compiler options and directives

Compilation of NAME is controlled through the use of compiler options and compiler directives. Compiler options are flags or settings for the compiler that influence its behaviour and the run-time behaviour of the executable that is produced. Options are specified when calling the compiler (e.g. in the compilation script or makefile) and they are often global in their influence. Different compilation options might be considered depending on the intended use. For example, a user will often create an optimised 'release' executable for general use but might also consider producing a 'debug' executable for testing purposes. Compiler directives are instructions embedded within the code, often intended to selectively execute a section of code (which might be compiler specific, for instance) or to control how a section of code is interpreted (such as parallelisation instructions). Specific details of compiler options and directives will be included in the [NAME DEVELOPER GUIDE](#).

1.8.3 Running NAME

NAME can be launched interactively from the command line (or DOS prompt on Windows systems) or it can be run indirectly through a script (under Linux) or batch file (under Windows). Most users tend to employ the script method as this enables other pre-processing and post-processing steps to be performed along with the NAME run itself. For instance, this might include the dynamic generation of the NAME input file (e.g., based on the current date or some other user-specified run information) and the post-run generation of graphical products. Scripting is also useful as a simple scheduling tool to launch a series of NAME scenarios one after the other. Indeed the basic concept of launching NAME indirectly through a script can be developed further, and the scripting can become more complex and interfaced with other software systems. For instance, use of NAME for operational emergency-response applications within the Met Office Operations Centre is through a web-based user interface, which can launch NAME on demand and then handle the generation and transmission of products. NAME has also been embedded as the dispersion modelling component within other tools and systems.

At the basic level of operation, NAME is invoked by launching its executable file (generically called `nameiii.exe`). This usually has a single argument associated with it, which is the filename of the main input file to be read by NAME. When running in interactive mode, if the input filename is not specified on the command line then the user will be prompted to supply an input file to be used. Further details on NAME input files are given in ‘[HEADED](#)’ [INPUT FILES FOR NAME](#) in 1.3.3.

Additional arguments can also be provided in the call to the NAME executable, although this is not especially common for most user applications. The main use for such arguments is associated with the restart functionality within NAME, although other aspects of the model run can also be controlled. See [COMMAND LINE ARGUMENTS](#) in 1.3.2 for further details.

There may be a requirement to set certain system or environment variables prior to launching the NAME executable. On Linux systems, for instance, the system stacksize limit often needs to be increased using the command **`ulimit -s unlimited`** (or some finite but large value, such as **`2000000`**, in place of **`unlimited`**). Similarly, the `OMP_STACKSIZE` environment variable should be set when running with multiple threads under OpenMP (see [PARALLELISATION OPTIONS](#) in 1.2.7.7). Other environment variables might also need to be set, for example, when linking in external shared object libraries or other applications such as the ECMWF GRIB-ABI. An advantage of using a script to launch NAME (over running NAME interactively via the command line) is that the relevant environment variable declarations can be included within the script.



A short practical introduction to running NAME for both Linux and Windows systems is supplied in the [» INSTALLATION AND TROUBLESHOOTING GUIDE «](#).

1.8.4 Parallelisation with OpenMP

1.8.4.1 Parallel computing – a quick primer covering the basic concepts

Computational problems are solved by a computer by constructing an algorithm (set of instructions) to solve the problem and then executing these instructions within a computer program. Traditionally software programs have been written based on the principle of serial computation in which the instructions are carried out as a serial stream of commands executed, one after another, on a central processing unit. In the serial computing paradigm, only one instruction can be performed at any one time. Once that instruction has completed, the next one is executed, and so on until the end of the program has been reached.

However it is often the case that large problems can be subdivided into a collection of smaller tasks which can be treated, more or less, independently of each other. By dividing the problem into independent parts, each part of the algorithm can be executed concurrently (i.e. simultaneously) with the other parts, enabling many calculations to be carried out 'in parallel'. In the parallel computing paradigm, multiple processing elements can be instructed to perform subtasks simultaneously and, in principle, the computer program should run faster overall.

Parallel computing has become standard practice on modern desktop and server architectures, where multiple processing elements are accessible either in the form of multi-core processors or through multiple processors within a single machine. A multi-core processor is a single computing 'chip' with multiple independent central processing units (CPUs), also known as 'cores'. These cores are the actual physical execution units on which the programming instructions are executed (here programming instructions are commands such as basic arithmetical and logical operations, and input/output communications). Computer processors were originally developed with only one core, but multi-core processors have become the norm in recent years – a dual-core processor has two cores, a quad-core processor contains four cores, etc.

A common feature of many multiple-processor systems is the concept of shared memory, in which a common area of random access memory (RAM) can be jointly accessed by the different processing units. A shared memory system offers several programming advantages since all processors share a single view of the data in memory and communications between processors can be very fast (although parallelisation always creates a small overhead as data needs to be exchanged between the different cores).

Notwithstanding that most modern computer platforms are now multi-processor systems, it is often a non-trivial task to implement programming instructions that are parallelised in an efficient and effective manner. Parallelisation of source code may require some careful considerations, e.g., to ensure that intended sub-tasks are independent, that data dependencies are handled correctly, and that suitable protection is applied when updating variables in shared memory. The OpenMP (Open Multi-Processing) framework supports shared memory multi-processing programming in Fortran (and also C and C++) by providing a multi-platform environment for developing parallelised code.

At a technical level, OpenMP is an API-based implementation of multi-threading for Fortran applications on shared memory systems. It consists of a collection of compiler directives, library routines and environment variables that influence run-time behaviour of the Fortran code. Preprocessor directives are embedded within the source code to provide task-sharing instructions to the 'master' thread (i.e. the primary process) to fork a specified number of slave threads whenever a task is to be distributed amongst them. These multiple threads then run concurrently on different processors until that parallelised section is completed, at which point the threads join back into the master thread and program execution continues as normal.

Phrase	Meaning
serial computing	processing instructions carried out consecutively one after another on a single CPU
parallel computing	processing instructions carried out concurrently using multiple CPUs
core	a single processing element that performs program instructions, also called a central processing unit (CPU)
multi-core processor	single computing 'chip' with two or more independent CPUs
thread	sub-task in a parallel program, sometimes called a process
multi-threading	parallelisation method where a task is divided among multiple threads
shared memory	common memory area accessible by several different CPUs
OpenMP	implementation of multi-threading for Fortran applications on shared memory multi-processing systems

Table 1.27: Some terminology associated with parallel computing

1.8.4.2 Parallelisation of NAME with OpenMP

Sections of the NAME code have been parallelised by using OpenMP directives to send the relevant parallelisation instructions to the lead processor. The behaviour of the parallel code at runtime is controlled by parameters supplied by the user in the [OpenMP Options](#) block in the main input file, see [PARALLELISATION OPTIONS](#) in 1.2.7.7 for details on how to set up a parallel NAME run.

As with most program codes, not all parts of NAME can be executed in parallel (or, at least, not efficiently). Parallelisation work has so far focussed on the following sections of code:

Particle Loop The loop over particles and puffs (and timesteps), which evolves all the particles and puffs over the period of time when they can be considered as independent entities. For most practical modelling applications, the particle loop parallelises in a highly efficient way with scaling that approaches 100 %.

Particle Update Loop The loop over particles that calculates the contribution of individual particles to the output requirements at synchronisation times.

Chemistry Loop The loop over the horizontal chemistry grid boxes for chemistry calculations.

Output Group Loop The loop over different groups of output requirements for writing output files.

Parallel MetRead A dedicated I/O thread can be launched for reading NWP met data files from disk in parallel to the main part of the processing performed by the other threads. This enables NWP met data to be read into memory ahead of schedule while the rest of the NAME run is progressing, and therefore for the meteorological fields to be immediately available to the program when required at the next time step.

The speed-up that can be expected by the user when allocating multiple threads to a NAME job can vary significantly. It depends on a range of factors, including the details of the NAME model set up, the underlying hardware on which NAME is being run, and the presence of any other processor- or I/O-intensive processes on that system. In general, running NAME on four processors does not necessarily make the code four times as fast!

In broad terms, the parallelisation speed-up depends on the relative time spent in those sections of the code that have been parallelised. Clearly there can be no speed-up from any parts of the code executed as a serial process. Code parallelisation therefore focussed deliberately on code sections that are typically expected to be computationally intensive. However, even in these sections, speed-ups are often sub-optimal, especially when there is extensive use of resources shared between all threads, such as disk access. Similarly, other processes run on the same machine may have an impact on the run time.

The general advice to users looking to optimise a NAME run on multiple threads is to first seek some initial guidance from us and to then experiment with the parallelisation settings. The user may find the NAME timer routines helpful in this regard, see [PROFILING AND THE USE OF TIMERS](#) in 1.8.5. These provide a simple utility to measure the time spent in different sections of the code.

Note that only one parallel I/O thread is allowed, as reading of the met data is likely to be limited more by disk access speed and internal data transfer rates rather than by processor throughput. So this section of the code does not scale in the traditional sense. Also note that it is currently not possible to use a parallel I/O thread together with the met-on-demand functionality.

Finally, a word of caution on the reproducibility of results from runs using multiple threads. As a consequence of the finite precision available on computer systems, floating point arithmetic is, in general, a non-associative operation. The value of a sum, product, etc. can depend on the order in which the individual values are combined. When running the parallel version of NAME, calculations can be split between the different threads and the order in which variables are updated can not be predicted. Therefore it is expected that the results between different parallel runs or between a parallel run and a serial run will differ slightly (even when the code has been compiled in debugging mode).

See [MD18/1: PARALLELISATION WITH OPENMP](#) for further details about NAME parallelisation.

1.8.5 Profiling and the use of timers

It can be useful for NAME users to understand where the program spends its execution time and especially to identify those parts of the program that consume the most time. Such information assists, for instance, when working to optimise the performance of NAME or in deciding on a suitable parallelisation strategy for a specific application. Various profiling tools are available to Fortran code developers in general and, in the case of NAME, the code supports use of the GNU profiler, GPROF, and the Intel performance tuning tool, VTUNE.

1.8.5.1 Profiling a program – the basics

Code profiling provides a means to learn where a program is spending its time and gives information on the execution call tree structure (i.e. how functions call other functions during execution of the program). This enables developers to identify slow sections of code or frequently called sections of code and so provides a focus for rewriting or restructuring those parts of the program in a more efficient way. A profiler uses information collected during the actual execution of a program and therefore only resolves features or aspects of the program that are active while the code is being profiled. In particular, it provides no information for any program options that are not used during the profiling run.

Profiling can generally be separated into three basic steps:

1. Compile and link the program with profiling enabled
2. Run the program to generate a profile data file
3. Run the profiler to analyze the profile data file

In the case of Fortran programs, two compiler options are relevant here:

-PG instructs the compiler to compile and link for function profiling with the GPROF tool

-G instructs the compiler to generate full debugging information in the object file (for use with VTune)

There is a computational overhead in generating profile information during program execution, and the profile data file can become large for complex programs. Profiling can also influence the compiler optimisation strategies (potentially switching off certain optimisations made by the compiler). Therefore it is generally recommended to only profile a program when the profiling information is actually required, rather than requesting this routinely as a default option.

1.8.5.2 Profiling NAME

NAME supports use of the GNU profiler, GPROF, (on Linux systems only) and the Intel performance tuning tool, VTUNE. The GNU profiler, GPROF, is reasonably straightforward to use and is provided

as freeware on most Linux platforms. VTUNE is an Intel performance tuning tool supplied as an add-on to their Fortran compiler package, and is designed for more extensive analysis and performance optimization of Fortran code. The simplest way to activate profiling options for NAME on Linux systems is to use the standard `Makefile` for source code compilation, see [COMPILING NAME](#) in 1.8.2.

The `Makefile` has two compiler options:

PROFILER=gprof sets the `-PG` option to compile and link for function profiling

VTUNE=true sets the `-G` option to generate full debugging information

See the user documentation on GPROF and VTUNE for further information on using these packages.

1.8.5.3 Running NAME with simple timers

As a simpler alternative to using a code profiler such as GPROF or VTUNE, there is also a rudimentary timing capability built into NAME. These ‘timers’ are designed to measure the time spent in individual sections of the model code, enabling the user to identify potential areas for optimisation or parallelisation. The sections of code that are currently ‘timed’ are listed in Table 1.28.

Code section	Description
WorkerThread_LPPATs	the main loop evolving particles and puffs over time steps
WorkerThread_CPRS	the particle update loop that calculates Lagrangian fields and sets of particle/puff information from particles
WorkerThread_ProcessAndOutputResults	the processing and output of results
ProcessFields	the processing of fields
OutputFields	the outputting of fields
WorkerThread#_NWPMetRead	reading NWP met data (where # is the thread number)
WorkerThread#_NWPMetProcess	processing NWP met data (where # is the thread number)
IOThread_NWPMetRead	reading NWP met data using a separate IO thread (when Parallel MetRead is enabled in OpenMP Options block)
IOThread_NWPMetProcess	processing NWP met data using a separate IO thread (when Parallel MetProcess is enabled in OpenMP Options block)
WorkerThread_ChemistryLoop	the main chemistry loop to update particles and chemistry fields
TotalTime	the total runtime (includes both timed and untimed regions)

Table 1.28: List of code sections that are timed in NAME.

To make use of the timer functionality in NAME, the source code needs to be compiled with the compiler directive `-DUseTimers` (see [COMPILATION OPTIONS AND DIRECTIVES](#) in 1.8.2.1). Furthermore, the timers will only work if OpenMP is also used (i.e. for ‘parallelised’ versions of NAME) because the timer module relies on a function in the OpenMP library. These compiler options are active by default when the standard `Makefile` is used on Linux systems.

The output of timing information is controlled by the **Timer Options:** block of the main input file. Timer information is requested by setting the variable **Use Timers?** to **Yes**. Note that the default setting is **not** to produce any timer information from a NAME run. However when it is requested, information from the timers is written to the timing log file `<InputFileStem>Timing.txt`, where `<InputFileStem>` is the stem of the main input file for the NAME run. For example, if the main input file is called `MyRun.txt`, then timer information is written to `MyRunTiming.txt`. The level of detail at which information is printed out is controlled by the keyword **Summary Only?**, which has a default value of **Yes** so as to only produce summary information at the end of the NAME run. By setting this option to **No**, detailed timing information is generated from each instance of a timed process. However the user should be aware that this can potentially create very large timing files.

Chapter 2

NAME – Supplementary Utilities

This second chapter takes a look at some supplementary utilities and programs that are not a core part of the dispersion model itself but which may be supplied with the NAME distribution (or are otherwise separately available as third-party downloads).

These utilities can be broadly divided into three categories: a) software associated with input/output of meteorological data, b) routines for creating graphical products from NAME output, and c) routines for converting NAME output to other formats.

2.1 Meteorological Utilities for NAME

NAME, as with any dispersion model, requires a source of meteorological input data to supply flow information for driving the dispersion simulation. This meteorological input might typically convey a local meteorology (as in the case of single-site met) or a gridded meteorology over a wider geographical area (such as NWP met). At the same time, meteorological and flow information can also be requested as user outputs from NAME. This section describes some utilities and software packages associated with the input and output of meteorological data in NAME.

NAME supports reading of NWP files in GRIB and NetCDF formats (in addition to the Met Office proprietary file formats, PP and NAME fieldsfiles). GRIB is a standard binary file format used by operational weather prediction centres for representing gridded met data. NetCDF is another common format used for representing weather and climate data. NAME is currently able to read its input meteorological fields from GRIB or NetCDF files, and there is a future intention to add an ability for NAME to produce outputs directly as GRIB or NetCDF files (although one of the issues here is that the conventions and standards are not always well-defined for dispersion model outputs).

2.1.1 Producing ADMS-style meteorology files with NAME

The single-site meteorology framework within NAME shares many characteristics with the treatment of meteorology in ADMS, including the design of the input met file format¹. Therefore, when referring to *ADMS met files* or *ADMS-style met files* in the following discussion, any statement applies equally to NAME single-site met files. One reason for referring to them here as ADMS met files is that they are a standard format recognised by the UK dispersion modelling community. A second reason is to assist the user to more clearly distinguish between a single-site met input file and other types of meteorology files associated with NAME.

NAME2ADMS is a file-conversion utility designed to convert output meteorology files from NAME into ADMS-style input met files. The utility is supplied as part of the NAME distribution as an executable file (and also as Fortran 90 source code, when source code is being provided with the distribution). The program is used in conjunction with NAME itself to produce single-site met files that can be used for dispersion modelling purposes (in NAME, ADMS or any other dispersion model recognising the ADMS file format).

A common situation where met files are generated in this way is in an *NWP met extraction run*, in which NAME is run using input meteorology from NWP fields to produce time series of selected weather variables at one or more specified locations. NAME will interpolate the NWP fields to the locations of interest and might also need to calculate some parameters if they are not directly supplied in the NWP input data set. In principle, any of the meteorological variables listed as flow quantities in Table 1.8 can be requested as outputs from a NAME run (and written as time-series data in a NAME output file). However the NAME2ADMS utility is designed to convert files containing a specific collection of meteorological variables, see Table 2.1. Specifically, it will generate an ADMS met file containing 13 data variables (including the date and time information). Any missing values in the NAME output file are automatically converted to the missing data indicator '-999.0' in the ADMS met file.

An example NAME input file for an NWP met extraction run is supplied in the `Runs` folder. The example run `Example_NWPMet_Extract` extracts hourly weather variables at two locations. Note that the output fields are in 'time series' format, and the fields at each location are directed into separate output files. Running this example would generate two NAME-style output files (a time-series file for each location) and each of these files would then need to be converted to an ADMS-style met file.

The NAME2ADMS utility program is run by calling its executable with two arguments – an *input file* and an *output file*. The input file is the NAME output file to be converted, and the output file is the ADMS-style met file that will be produced. The utility reads the required met fields from the NAME output file and writes the information in ADMS-style time series format to the output file. Note that in practice the executable can be run from the command line or from within a script.

¹A single-site met file for NAME has the same basic structure as an ADMS met input file, but there are some differences between the two modelling systems in terms of the met parameters that are recognised. For instance, NAME supports a time-zone option indicating how input times are to be interpreted (ADMS considers its times to be referenced to local solar time).

Field Request in NAME	ADMS-style Met Parameter
[Time]	YEAR
	MONTH
	DAY OF MONTH
	DAY
	HOUR
Wind Speed	WIND SPEED
Wind Direction (degrees)	WIND DIRECTION (DEGREES)
Cloud Amount (oktas)	CLOUD AMOUNT (OKTAS)
Temperature (C)	TEMPERATURE (C)
Sensible heat flux	SENSIBLE HEAT FLUX
Boundary layer depth	BOUNDARY LAYER DEPTH
Precipitation rate (mm/hr)	PRECIPITATION RATE (MM/HOUR)
Relative Humidity (%)	RELATIVE HUMIDITY (PERCENT)

Table 2.1: List of meteorological parameters processed by the NAME2ADMS utility

Another scenario where NAME and NAME2ADMS might be useful is where an existing single-site met file contains only a limited collection of weather variables but the user would appreciate (estimates of) other variables as well. For instance, observations from a weather station are often limited to a basic set of measurements of wind, temperature, cloud and precipitation and would not include measurements of surface heat flux or indicators of stability. In this situation, the NAME single-site met preprocessor can be run on the available meteorological inputs to estimate values for missing meteorological parameters. An updated single-site met file can be generated for analysis purposes or further use in dispersion modelling applications subject, of course, to the caveat that they are only estimates based on various assumptions and approximations that will be subject to uncertainties (but there are also uncertainties within the observations, of course!).

2.1.2 Reading GRIB format meteorological data

GRIB (GRIdded Binary) is a standard binary file format used by operational weather prediction centres for the transmission and storage of gridded met data fields. As far as NAME is concerned, GRIB formatted met files are most frequently encountered with the meteorological data sets obtained from the European Centre for Medium-Range Weather Forecasts (ECMWF). This includes real-time analyses and forecasts (deterministic and ensemble) as well as reanalysis data sets such as ERA-40 and ERA-Interim.

NAME supports *reading* of NWP met files in GRIB through calls to the ECMWF GRIB API. The ECMWF GRIB API is an Application Program Interface accessible from C and Fortran programs that enables the encoding and decoding of WMO FM-92 GRIB edition 1 and edition 2 messages.



The ECMWF GRIB API is only supported on certain platforms (these are mostly Linux-based systems). The option to build a NAME executable with support for the GRIB API is currently only available under Linux. This facility is not available to users on Windows systems.

When there is a user requirement to read NWP met files in GRIB format, the following steps should be performed when installing the NAME distribution:

1. Compilation and installation of the GRIB API (if not already installed on the system). This ensures that the required shared libraries, tables, etc. are available to NAME at run time.
2. Create a build of NAME with support for the GRIB API. This produces a special form of the NAME executable which includes extended functionality for dynamically calling the relevant GRIB API routines when reading NWP met files in GRIB format. An executable supporting dynamic linking with the GRIB API is identified by the inclusion of the qualifier 'GribAPI' in its filename (e.g. `nameiiiGribAPI_64bit_par.exe`).

The ECMWF GRIB API is supplied as an additional component to the NAME distribution, see the top-level `ReadMe.txt` for instructions on installing the GRIB API for use with NAME from the supplied distribution. Alternatively, the latest version of the ECMWF GRIB API software can be downloaded directly from the ECMWF website at

<https://software.ecmwf.int/wiki/display/GRIB/Releases>

Full documentation on the GRIB API software can be found via the links

<https://software.ecmwf.int/wiki/display/GRIB/Home>

<https://software.ecmwf.int/wiki/display/GRIB/Documentation>

The ECMWF GRIB API software is licensed under the GNU Lesser General Public License, which incorporates the terms and conditions of version 3 of the GNU General Public License.

A few local modifications (including addition of local tables) are contained in the version supplied with the NAME distribution, see `Code_GribAPI/Changes.txt` for details.

- A standard compile script has been added to automate the execution of the compilation and installation process for the ECMWF GRIB API on Linux systems.
- Local definition files have been included describing the meteorological parameters available in processed ECMWF datasets for NAME.

These modifications can be provided to other users on request.

The configuration script supplied as part of the standard software package will automatically determine the operating system, available compilers, etc. and generate an appropriate makefile for the compilation step. Source files are compiled into the shared object libraries `libgrib-api.so`,

`libgrib_api_f77.so` and `libgrib_api_f90.so` and then installed, along with tables and templates, into the folder `SharedLibraries_Linux`. Note that `‘_64bit’` is added to the names of the libraries when compiled on 64-bit systems. The shared libraries, tables and templates are then available for dynamic linking into NAME at run time. See the `README` file in the `grib_api-X.X.X` package for more details on compilation, etc.



A pre-built NAME executable supporting dynamic linking with the ECMWF GRIB API at run time is supplied with the NAME distribution. However to ensure consistency of library references, etc. on any specific system, users intending to invoke NAME with GRIB API functionality are advised to recompile the NAME executable on their system, see [COMPILING NAME](#) in 1.8.2.

At run-time, the NAME executable has to link to the GRIB API libraries and be able to find the relevant tables, etc. NAME needs to be told where to look for these files on the system, which is achieved by setting various environment variables prior to launching the executable. The following environment variables should be defined and exported explicitly on the command line (or the same can be done through a script):

- `export GRIB_DEFINITION_PATH=${SHAREDLIB_DIR}/share/definitions`
- `export GRIB_SAMPLES_PATH=${SHAREDLIB_DIR}/share/samples`
- `export GRIB_API_INCLUDE=${SHAREDLIB_DIR}/include`
- `export GRIB_API_LIB=${SHAREDLIB_DIR}/lib`

where `${SHAREDLIB_DIR}` is the top-level directory of the GRIB API installation. Additionally, the path `${GRIB_API_LIB}` might need to be added to the `LD_LIBRARY_PATH` environment variable if the GRIB API is a non-standard library on your system. This environment variable lists a colon-separated set of directories which the program loader will use to search for libraries (before it searches the standard system libraries).

The environment variable `GRIB_DEFINITION_PATH` can be set to use local definition files in preference to the standard definition files provided within the GRIB API distribution. To do this, set the environment variable as

```
export GRIB_DEFINITION_PATH=
    ${SHAREDLIB_DIR}/local/definitions:${SHAREDLIB_DIR}/share/definitions
```

where, as above, `${SHAREDLIB_DIR}` is the top-level directory of the GRIB API installation and the local definition files are stored under the `/local` subdirectory.

Users with an interest in using GRIB format data files should contact us for further information. Note that the Met Office will not normally supply ECMWF meteorological data sets, although we may be able to provide advice on retrieving data directly from the ECMWF MARS database.

2.1.3 Reading NetCDF format meteorological data

NetCDF (Network Common Data Form) is a set of software libraries and self-describing, machine-independent data formats that support the creation, access, and sharing of array-oriented scientific data. Further information on NetCDF can be found via the Unidata website at:

<http://www.unidata.ucar.edu/software/netcdf/>

NAME supports *reading* of NetCDF met files via calls to the NetCDF API. This software package provides Fortran application interfaces for accessing netCDF data. It depends on the netCDF C library, which must be installed first, which in turn depends on HDF5 and zlib libraries. The NetCDF API is not supplied with the NAME distribution, but is available separately as third-party software released under the Apache 2.0 Open Source License. See the top-level [ReadMe.txt](#) for further details on the license terms and for instructions on downloading and installing the software package.

When there is a user requirement to read input NWP met files in NetCDF format, the following steps should be performed when installing the NAME distribution:

1. Check that there is a NetCDF library installed on the system you are running on. This ensures that the required shared libraries are available to NAME at run time. Instructions on how to download and install the library are given in the top-level [ReadMe.txt](#) file.
2. Edit the [Makefile](#) to set the directories to be searched for the NetCDF API library files. Note that the default search paths are configured for the Met Office Linux desktop estate and no changes are needed here for research users within the Met Office.

- (a) Set the directory passed to the compilation in the `-I` option of the `CFLAGS` variable. This indicates the directory to be searched for NetCDF header files.

e.g., `CFLAGS += -DNetCDFsupport`
`-I/usr/local/sci/include/netcdf_fortran/ifort_compos`

- (b) Set the directory passed to the linker in the `-L` option of the `LFLAGS` variable. This indicates the directory to be searched for the NetCDF Fortran shared object library file.

e.g., `LFLAGS += -L/usr/local/sci/lib/netcdf_fortran/ifort_composerxe`
`-lnetcdff`

3. Create a build of NAME with support for reading NetCDF data files using the NetCDF API. This is achieved by including the option `USENETCDF=true` in the `MAKE` compilation, see [COMPILING NAME](#) in 1.8.2. Compiling NAME in this way will activate the appropriate calls to the NetCDF library routines, producing a special form of the NAME executable which includes extended functionality for dynamically calling the relevant NetCDF library routines when reading NWP met files in NetCDF format. An executable supporting dynamic linking with the NetCDF API

is identified by the inclusion of the qualifier 'NetCDF' in its filename (e.g. `nameiiiNetCDF_-64bit_par.exe`). Note that this executable may also be used in the usual way with PP files from the UM.

At run-time, the NAME executable links dynamically to the NetCDF shared libraries. If the NetCDF libraries are not in the standard search path on your system, then NAME needs to be told where to look for these files. This is achieved by including the directory that contains the shared library `libnetcdf.so` within the `LD_LIBRARY_PATH` environment variable. This environment variable lists a colon-separated set of directories which the program loader will use to search for libraries (before it searches the standard system libraries).

An extra variable **NC Field Name** is required in the **NWP Met File Structure Definition** input block when using met data in NetCDF format. This should contain the names of the meteorological variables as they appear in the NetCDF files. An example of a met definition file for use with NetCDF data can be supplied on request. Furthermore, in instances where NWP data is being read in NetCDF format, the corresponding topography file also needs to be in NetCDF. NAME expects the NetCDF name of the variable in the topography file to be 'topography'. The variable name in the NetCDF topography file can be changed, if necessary, using the NCO command `ncrename -v`.

Users with an interest in using NetCDF format data files should contact us for further information.

2.2 Producing Graphics from NAME Output

All output data files created by the NAME model are in a plain text format. Although a user can inspect the raw data directly, and indeed might decide to use some of the simpler output requests in this way, it would be more usual for output files to be processed in some manner. Processing of the output files typically involves the generation of graphical products, although other forms of data analysis or statistical manipulation are also sometimes performed.

This section presents a brief overview of IDL and Python plotting routines that are available for NAME. These routines have been written by the Met Office and are distributed as part of the NAME package. Users are free to use the supplied routines, or they may wish to develop their own utilities for processing and plotting NAME outputs instead. NAME users are encouraged to feedback their comments on the standard routines supplied and also to share any new utilities that are developed when this might be beneficial to the wider community.

2.2.1 Generating graphics using IDL

Currently the standard approach to generate graphics from NAME output files uses the IDL visualisation software (see <http://www.exelisvis.com/>). A set of IDL routines is supplied with NAME in the subfolder `Code_IDLGraphics` and they can be used for plotting NAME data stored in the most common styles of output file (including the legacy NAME II output formats).

Three main plotting routines are available: `plotfield.pro`, `plotseries.pro` and `plottraj.pro`, which are called to plot field data, time-series data and trajectory data, respectively. See [FORMAT OF NAME OUTPUT FILES](#) in 1.4.4 for further details on the types of output files that are supported for plotting. A brief description of each of these routines is given below. The `Code_IDLGraphics` subfolder also contains various other plotting and analysis procedures for use with NAME outputs.

`plotfield.pro` Routine for plotting 2-dimensional (x, y) field data on a regular horizontal grid. Output is produced in PostScript format using an A4 page size, although it is also possible to produce images in GIF (Graphics Interchange Format) and JPEG (Joint Photographic Experts Group) formats. The routine requires two mandatory arguments to run: the full path of the directory in which the data are stored, and the name identifying the output group to which the fields belong. The user is also able to choose from a large selection of plotting options, e.g., to filter on the data fields to be plotted, to set the geographical area of the plot, or to set the contouring intervals and colours to be used. A full list of available options can be found in the comments section at the start of the script.

`plotseries.pro` Routine for plotting time series of air concentration, deposition or meteorological parameters. By default, the data are plotted on a logarithmic y-axis but the user can request a linear y-axis if needed. See the script for a full list of the available options.

`plottraj.pro` Routine for plotting particle trajectories. The routine will plot trajectories from a set of particle/puff information identified by its output group. All trajectories must cover the same time period.

Examples of the three types of graphical products are illustrated in Figures 2.1, 2.2 and 2.3.

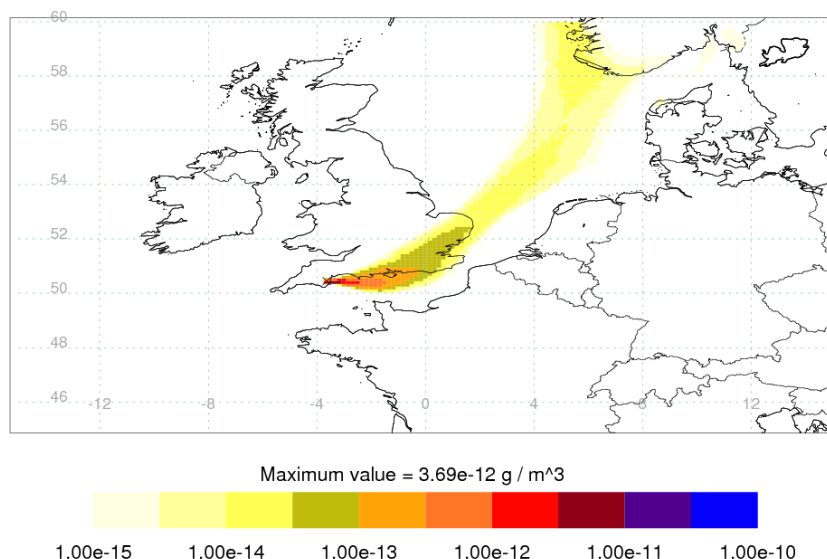


Figure 2.1: Example of output from the IDL routine `plotfield.pro`

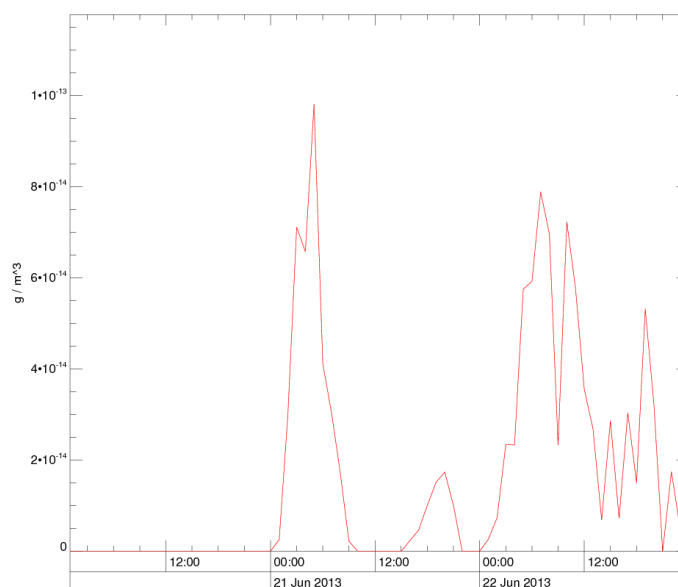


Figure 2.2: Example of output from the IDL routine `plotseries.pro`

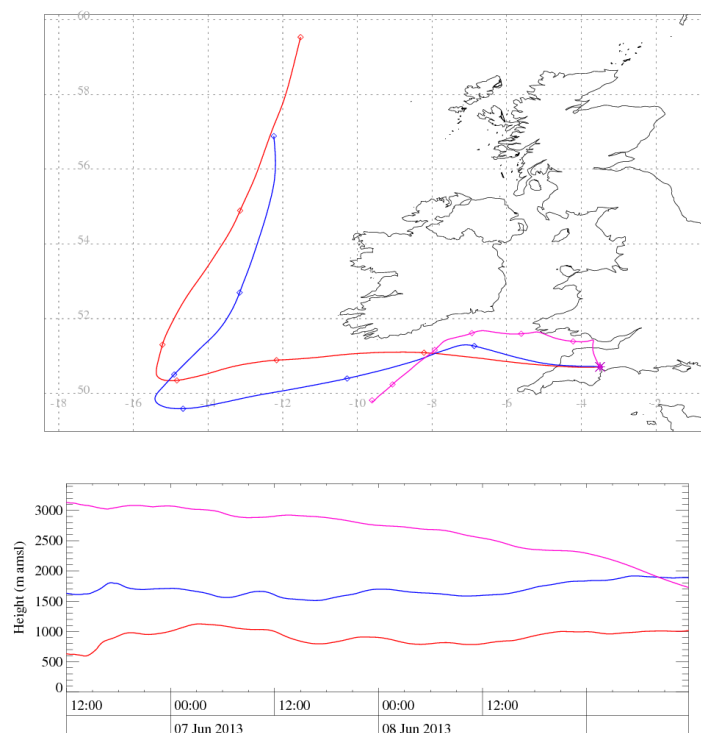


Figure 2.3: Example of output from the IDL routine `plottraj.pro`

2.2.2 Generating graphics using Python

The Met Office is migrating towards the use of Python as a visualisation software platform, and has developed an open-source software package called IRIS as a tool to aid data analysis and plotting (see <http://scitools.org.uk/iris/index.html> for further details and instructions on downloading IRIS). Starting with NAME Version 6.3, a set of Python routines is supplied with NAME in the subfolder `Code_Python` to allow a user to load NAME data using IRIS and thereby give the user access to the plotting and manipulation tools built into IRIS. The intention is that these routines for loading NAME data will be incorporated directly into a future release of IRIS. Python plotting routines specific to NAME are also being developed and further details will appear in the documentation once these have become more established.

2.3 Post-processing of NAME Output

At the present time, all NAME output is produced in ASCII plain text, which is formatted according to the descriptions given in [FORMAT OF NAME OUTPUT FILES](#) in 1.4.4. However a longer term aim is to extend NAME functionality to support direct output in other (binary) data formats. In particular, there is an intention to add a capability to generate output files in GRIB and NetCDF formats as these two formats are commonly used within the meteorological and dispersion modelling communities for the representation of environmental data. For users requiring output in GRIB or NetCDF formats in the interim, it is possible to convert existing text-based output files from NAME into these other formats via a post-processing step.

2.3.1 Converting NAME output to GRIB format

NAME cannot create output files in GRIB format directly, however a utility is available to convert NAME output text files to GRIB 2 format as a post-processing activity. The Fortran program reads a NAME output file, and extracts the field values and various metadata information such as the parameters defining the grid structure. It then uses routines in the ECMWF GRIB API library to encode each two-dimensional NAME output field as a GRIB 2 message. The capability was developed several years ago for a specific project requiring the visualisation of NAME concentration fields and so the utility has been designed to read a particular set of NAME output files. However the approach should, in principle, be suitable for the conversion of general NAME output fields (although users might need to modify the software code to suit their own NAME output file structure).



The utility for converting NAME output to GRIB2 was developed based on a specific release of the ECMWF GRIB API, and it cannot be guaranteed that it will function without modification for other versions of the GRIB API.

2.3.2 Converting NAME output to NetCDF format

Although NAME cannot directly output to NetCDF files, tools have been developed in both Fortran and Python for converting NAME output files to NetCDF format. The Python script is included in the [Code_Python](#) folder of the NAME distribution for external users, but the Fortran code can also be supplied on request. Currently these routines only generate NetCDF files from NAME output files containing horizontally gridded fields. All the routines generate 'Classic' NetCDF files although the python code may be modified to produce 'NetCDF version 4' NetCDF files.

Three Fortran routines exist:

[name2_to_cdf](#) Reads a NAME II format fields file and writes out all fields to a NetCDF file. Fields are assigned names based on their order in the file, being labelled as *field01*, *field02*, etc.

`name_to_cdf_field` Reads a NAME III format fields file and writes out all fields to a NetCDF file. Fields are assigned names based on their order in the file, being labelled as *field01*, *field02*, etc.

`name_to_cdf` Reads a NAME III format fields file and writes out all fields to a NetCDF file. This version assigns names to the fields using the NAME attribute in the field column headers (which needs to be suitably specified within the output requests block when requesting the field). Field names should not contain any spaces, and a unique name must be assigned to each output field.

A single python routine also exists:

`NAME2NetCDF_inclload.py` This routine can handle both NAME II and NAME III format fields files.

Chapter 3

NAME – Further Aspects

3.1 Documentation

An extensive collection of NAME model documents is included within the NAME distribution, see Table 3.1. These model documents serve a variety of needs – they provide further information on using the NAME model, technical descriptions of various aspects of NAME, as well as model validation reports (see also [NAME TESTING AND VALIDATION](#) in 3.2). The NAME documentation set is being gradually extended and will be revised on a regular basis.

Category	Document Ref.	Subject	Full Description
Dispersion	MD1/1	Puffs	Description of puff concept
	MD1/2	Dry Deposition	Description of land-use dependent dry deposition scheme
	MD1/3	Deep Convection	Parametrization of deep convection in NAME III
	MD1/4	Eulerian Model	Eulerian-Lagrangian hybrid model
Model structure	MD2/1	Code Structure	Code design, structure and conventions
	MD2/2	Input	Model inputs for NAME
	MD2/3	Errors	Explanation of the more common NAME error messages
	MD2/4	Output	Description of NAME fields output format
Basic concepts	MD3/1	Coordinates Investigation	Investigation of accuracy of latitude-longitude coordinate transformations
	MD3/2	Coordinates	Horizontal and vertical coordinate systems supported by NAME
	MD3/3	Constants	Physical and mathematical constants and functions
	MD3/4	Unit Conversions	Conversion of units for output field requests
Radiological	MD4/1	Decay Chains	Implementation of decay chain modelling
	MD4/2	Cloud Gamma	Implementation of cloud gamma modelling
Fluctuations	MD6/1	Fluctuations	Scheme for predicting concentration fluctuations
Buildings	MD8/1	Buildings	Stand-alone model for dispersion around an isolated cuboid-shaped building
	MD8/2	Buildings Tests	Preliminary testing of the stand-alone building effects model
Terrain	MD9/1	Lincom	Use of the LINCOM linear flow model in NAME
Rainfall Data	MD12/1	Radar Rainfall	Use of radar rainfall data in NAME

Table 3.1: List of NAME model documents (Part 1: NAME science and model components)

Category	Document Ref.	Subject	Full Description
Validation	MD13/1	Kincaid	Validation against the Kincaid power plant dataset
	MD13/2	Etna	Validation against the October 2002 volcanic eruption of Mount Etna
	MD13/3	Hekla	Validation against the February 2000 volcanic eruption of Hekla
	MD13/4	Basic Tests	Basic consistency checks (e.g. comparing particles v puffs)
	MD13/5	Deposition	Basic checks of deposition
	MD13/6	High-resolution UM	Test case exploring the effect of NWP resolution
	MD13/7	PUMA I	Comparison against PUMA summer campaign using original chemistry scheme
	MD13/8	PUMA II	Updated comparison against PUMA campaign with revised chemistry scheme
	MD13/9	Uniform Field	Investigating abilities to maintain a uniform field of concentration
	MD13/10	Nottingham	Validation against September 1998 sulphur dioxide pollution episode in Nottingham
	MD13/11	Ozone Case Study	Understanding the effect of the land use dry deposition scheme on predicted ozone concentrations
Pre-existing components	MD14/1	NAME Porting	Early porting work to Fortran 90
User guidance and training	MD15/1	Introduction For Users	Short introductory presentation and example input files
	MD15/2	Introduction For Developers	Short introductory presentation for model developers
	MD15/3	Computer Requirements	Technical requirements for NAME (<i>deprecated – replaced by 15/6</i>)
	MD15/4	Licensing	Summary of NAME licensing (<i>needs updating</i>)
	MD15/5	List Of Papers	List of peer-reviewed science papers for NAME (<i>needs updating</i>)
	MD15/6	Installation	Installation and trouble-shooting guide
	MD15/7	User Guide For NWP	User Guide for running NAME with NWP Met Data
	MD15/8	User Guide	Main User Guide for NAME
Meteorology	MD17/1	Global-Cutout Domains	Description of 'cutout' regions for Global UM met data
	MD17/2	Met Data	Technical information on UM met data for NAME (<i>deprecated – replaced by 15/7</i>)
Coding and performance	MD18/1	Parallelisation	Description of NAME parallelisation with OpenMP

Table 3.1: List of NAME model documents, cont. (Part 2: NAME validation and user guides)

3.2 Testing and Validation

Model testing and validation are integral components in the development process for any scientific model. The former set of activities are concerned, broadly speaking, with ensuring that the model performs in agreement with its intended design (i.e. that there are no coding errors leading to erroneous calculations), whereas the latter activities aim to assess how well the model results agree with measurements or other observational evidence of real situations. Both types of activity are regularly conducted with NAME, with code testing being performed during any model development work and opportunities continuously sought for suitable validation studies against which NAME might be assessed.

Some basic code testing is reported in some of the early model documentation papers (such as ensuring that dispersion predictions based on the puff dispersion scheme are in general agreement with those obtained using particles). Most of the model testing, however, is carried out by NAME developers as part of the code development process and results are recorded in the change management documentation (e.g. the NAME 'Changes' file and, more recently, our FCM log messages). Comprehensive testing of NAME is conducted whenever a stable release (that is, 'frozen' version) of the model is produced.

The most comprehensive NAME validation study to date has been against the Kincaid data set, using a process based on the Model Validation Kit methodology developed under the Harmonisation initiative. NAME results for Kincaid compare favourably with other leading atmospheric dispersion models. NAME has also been validated against the ETEX long-range tracer experiment, and our intention is to perform further validation activities against standard dispersion datasets, such as Project Prairie Grass, ANATEX, CAPTEX and DAPPLE, as and when time permits.

Individual case studies are also regularly identified for qualitative, and sometimes quantitative, comparison against observational data. For instance, suitable test cases may include volcanic eruptions, dust storms, forest fires, industrial fires or nuclear releases. The Eyjafjallajökull eruption of April/May 2010 and Fukushima nuclear accident of March/April 2011 are two recent examples.

3.3 NAME Licensing

NAME is available under licence for **non-commercial research use**. Wider licensing terms are currently being developed for operational production and commercial usage of NAME, and the intention is that NAME will become available for these wider uses in the near future. Please contact the Met Office with any NAME licensing enquiries or to obtain a copy of the latest licensing terms and conditions.

Note that meteorological data sets for use with NAME are licensed under separate arrangements.

3.4 Installation and Model Upgrades

The NAME distribution can be downloaded via a dedicated FTP account on the Met Office research FTP server. Full instructions for accessing the FTP server and retrieving the relevant zip file(s) are provided as required. Each zip file is password-protected for security. On request, NAME can also be supplied on a DVD.

The standard NAME distribution includes NAME executables (compiled for various platforms), supporting resources, example runs and model documentation. The source code might also be provided for some users. See [» MD15/6: INSTALLATION AND TROUBLESHOOTING GUIDE «](#) for further details on installing the NAME distribution on your system.

Upgrades to newer versions of NAME typically occur every 6 to 12 months and are made available by the same route (i.e. FTP download or, exceptionally, by DVD). Incremental updates such as critical bug fixes might also be provided via e-mail or other electronic means.

3.5 NAME Training

The Met Office can provide training and guidance on the use of NAME, and this is recommended especially for new users of the model. A NAME Training Package has been developed consisting of a two-day taught course¹, a self-guide tutorial and up to three days of post-course support. Additional support may be provided at the discretion of the Met Office for which an extra charge may be incurred.

The two-day training course is flexible to the requirements of individual customers. The training style is kept deliberately informal to encourage interaction and discussions. The first day usually involves a classroom-style set of presentations that are designed to introduce the user to NAME and its broad range of applications, and also provide a basic understanding of how to set up a model run. The second day builds on this foundation by giving users an opportunity to run NAME for themselves by working through the tutorial examples, and to have more detailed discussions concerning specific aspects of NAME that are relevant to their intended applications. This might also include, for instance, discussion of suitable sources of meteorological data for running NAME.

Please contact the Met Office with any enquiries about NAME training.

¹Courses are usually held at the Met Office in Exeter, but on request can be delivered at a customer's own premises provided that suitable training facilities are available. Note that an additional cost will be incurred to cover T & S expenses for any visits of Met Office staff to another site.

3.6 NAME User Community

A 'user community' has been established amongst NAME users and its activities are gradually being developed. The intention is that it should act as both a formal method of communication and an informal means of discussion between external NAME users (e.g. a self-help group).

The NAME User Community 'lives' on the Met Office research collaboration Twiki, see

<http://collab.metoffice.gov.uk/twiki/bin/view/NAME/WebHome>

Users first need to register to use the website. Registration is a two-stage process.

1. Requests for a new account on the collaboration website should be sent to us by email. The email should contain the name of the individual requesting the account and their official email address (this address will be identified with their account and will be used for account administration purposes). *Note that access to the website is subject to terms and conditions and can only be granted to members of academic institutions and public bodies.*
2. Following receipt of a request, the user will be pre-registered on the system. Once this has been done, further instructions will be sent to the new user by email, instructing them on how to complete the online registration process.

Although the above sounds complicated, in practice registration is quick and easy to complete!

3.7 NAME Bibliography

A concise bibliography of NAME papers arranged by different subject headings is presented here. A more extensive list of NAME literature is available on request.

General references and description of NAME physics

General

- Meneguz, E. and Thomson, D.J., 'Towards a new scheme for parametrisation of deep convection in NAME III', *Int. J. Environ. and Pollution*, **54**, pp. 128-136, 2014.
- Mueller, E.H., Ford, R., Hort, M.C., Huggett, L., Riley, G. and Thomson, D.J., 'Parallelisation of the Lagrangian atmospheric dispersion model NAME', *Comp. Phys. Comms.* **184**, pp. 2734-2745, 2013.
- Thomson, D.J. and Jones, A.R., 'A new puff modelling technique for short range dispersion applications', *Int. J. Environ. and Pollution* **44**, pp. 156-163, 2011.
- Jones, A.R., Thomson, D.J., Hort, M. and Devenish, B., 'The U.K. Met Office's next-generation atmospheric dispersion model, NAME III', In: *Air Pollution Modeling and its Application XVII (Proceedings of the 27th NATO/CCMS International Technical Meeting on Air Pollution Modelling and its Application)*, Springer, Borrego C. and Norman A.L. (Eds.), pp. 580-589, 2007.
- Jones, A.R., 'A fluctuations scheme in NAME III', NAME III Documentation, MD6/1, 2003.
- Thomson, D.J., 'Dispersion of particle pairs and decay of scalar fields in isotropic turbulence', *Physics of Fluids* **15**, no.3, pp. 801-813, 2003.
- Thomson, D.J. and Manning, A.J., 'Along-wind dispersion in light wind conditions', *Boundary-Layer Meteorology* **98**, pp. 341-358, 2001.
- Maryon, R.H., Ryall, D.B. and Malcolm, A.L., 'The NAME 4 Dispersion Model: Science Documentation', Turbulence and Diffusion Note 262, Met Office, UK, 1999.

Random walk models

- Thomson, D.J., Physick, W.L. and Maryon, R.H., 'Treatment of interfaces in random walk dispersion models', *Journal of Applied Meteorology* **36**, pp. 1284-1295, 1997.
- Thomson, D.J. and Montgomery, M.R., 'Reflection boundary conditions for random walk models of dispersion in non-Gaussian turbulence', *Atmospheric Environment* **28**, pp. 1981-1987., 1994.
- Thomson, D.J., 'Criteria for the selection of stochastic models of particle trajectories in turbulent flows', *J. Fluid Mech.* **180**, pp. 529-556, 1987.

Turbulence

- Webster, H.N., Thomson, D.J. and Morrison, N.L., 'Parametrizing low-frequency mesoscale motions in atmospheric dispersion models', In: Proceedings of the 16th international conference on harmonisation within atmospheric dispersion modelling for regulatory purposes, 2014.
- Morrison, N.L. and Webster, H.N., 'An assessment of turbulence profiles in rural and urban environments using local measurements and NWP results', *Boundary-Layer Meteorol.* **115**, pp. 223-239, 2005.
- Webster, H.N. and Thomson, D.J., 'Parameterising low-frequency meander in atmospheric dispersion models', In: Tenth international conference on harmonisation within atmospheric dispersion modelling for regulatory purposes, Skouloudis, A.N., Kassomenos, P. and Bartzis, J. (Eds.), pp. 594-598, 2005.
- Webster, H.N., Thomson, D.J. and Morrison, N.L., 'New turbulence profiles for NAME', Turbulence and Diffusion Note 288, Met Office, UK, 2003.

Deposition

- Leadbetter, S.J., Hort, M.C., Jones, A.R., Webster, H.N. and Draxler, R.R., 'Sensitivity of the modelled deposition of Caesium-137 from the Fukushima Dai-ichi nuclear power plant to the wet deposition parametrisation in NAME', *Journal of Environmental Radioactivity* **139**, pp. 200-211, 2015.
- Webster, H.N. and Thomson, D.J., 'The NAME wet deposition scheme', Met Office Forecasting Research Technical Report 584, Met Office, UK, 2014.
- Webster, H.N., Thomson, D.J. and Jones, A.R., 'Modelling wet deposition with high resolution precipitation data', In: Proceedings of the 15th international conference on harmonisation within atmospheric dispersion modelling for regulatory purposes, eds. R. San Jose and J.L. Perez, pp. 378-381, 2013.
- Webster, H.N. and Thomson, D.J., 'A land-use dependent dry deposition scheme for NAME', NAME III Document, MD1/2, 2012.
- Webster, H.N. and Thomson, D.J., 'Dry deposition modelling in a Lagrangian dispersion model', *Int. J. Env. Pollution* **47**, pp. 1-9, 2011.

Plume rise

- Webster, H.N., Devenish, B.J., Haywood, J.M., Lock, A.P. and Thomson, D.J., 'Using plume rise schemes to model highly buoyant plumes from large fires', *International Journal of Environment and Pollution (IJEP), Special Issue: 11th International Conference on Harmonisation*

within Atmospheric Dispersion Modelling for Regulatory Purposes, Guest Editors: Dr. David Carruthers and Dr. Christine McHugh, bf 44, pp. 226-243, doi: 10.1504/IJEP.2011.038422, 2011.

Devenish, B. J., Rooney, G. G. and Thomson, D. J., 'Large-eddy simulation of a buoyant plume in uniform and stably stratified environments', *J. Fluid Mech.* **652**, pp. 75-103, 2010.

Devenish, B.J., Rooney, G.G., Webster, H.N. and Thomson, D.J., 'The Entrainment Rate for Buoyant Plumes in a Crossflow', *Boundary-Layer Meteorol.* **134**, pp. 411-439, 2010.

Webster, H.N. and Thomson, D.J., 'In search of a new plume rise scheme for NAME', Turbulence and Diffusion Note 272, Met Office, UK, 2001.

Validation

Derwent, R., Simmonds, P. G., Manning, A. J., O'Doherty, S., and Spain, G., 'Methane emissions from peat bogs in the vicinity of the Mace Head Atmospheric Research Station over a 12-year period', *Atmospheric Environment*, doi: 10.1016/j.atmosenv.2009.01.026, 2009.

Jones, A.R., 'Testing the NAME III chemistry scheme - updated comparisons with NAME II predictions for the PUMA summer campaign', NAME III Documentation, MD13/8, 2007.

Jones, A.R., 'NAME III modelling of the Nottingham sulphur dioxide pollution episode in September 1998', NAME III Documentation, MD13/10, 2007.

Witham, C.S., Hort, M.C., Potts, R., Servranckx, R., Husson, P. and Bonnardot, F., 'Comparison of VAAC atmospheric dispersion models using the 1 November 2004 Grimsvötn eruption', *Meteorological Applications* **14**, 27-38, 2007.

Jones, A.R., Webster, H.N., Hort, M.C., Thomson, D.J., 'Validation of NAME III (Version 2.0) against the Kincaid data set', NAME III Documentation, MD13/1, 2005.

Jones, A.R., Redington, A.L., 'Testing the NAME III chemistry scheme by a comparison of the NAME II and NAME III model predictions for the PUMA summer campaign', NAME III Documentation, MD13/7, 2005.

Hort, M.C., 'NAME III (v1.3) Validation: Etna October 2002 Eruption', NAME III Documentation, MD13/2, 2003.

Hort, M.C., 'NAME III (v1.3) Validation: Hekla February 2000 Eruption', NAME III Documentation, MD13/3, 2003.

Webster, H.N. and Thomson, D.J., 'Validation of a Lagrangian model plume rise scheme using the Kincaid data set', *Atmos. Environ.* **36**, pp. 5031-5042, 2002.

Manning, A.J., 'Predicting NO_x levels in urban areas using two different dispersion models', In: *International Journal of Environment and Pollutions*, conference proceedings of the *Sixth*

international conference on harmonisation within atmospheric dispersion modelling for regulatory purposes, Rouen, France, 1999.

Ryall, D.B. and Maryon, R.H., 'Validation of the UK Met Office's NAME model against the ETEX dataset', *Atmos. Environ.* **32**, pp. 4265-4276, 1998.

Chemistry

Redington, A.L. and Derwent, R.G., 'Modelling secondary organic aerosol in the United Kingdom', *Atmos. Environ.* **64**, pp. 349-357, 2013.

Redington, A.L., Derwent, R.G., Witham, C.S. and Manning, A.J., 'Sensitivity of modelled sulphate and nitrate aerosol to cloud, pH and ammonia emissions', *Atmos. Environ.* **43**, pp. 3227-3234, 2009.

Middleton, D.R., Ordonez, C. and Savage, N., 'Comparison of modelled chemistry in AQUM and NAME III air quality forecasts with AURN monitoring data', 2009.

Middleton, D.R., Jones, A.R., Redington, A.L., Thomson, D.J., Sokhi, R.S., Luhana, L. and Fisher, B.E.A., 'Lagrangian modelling of plume chemistry for secondary pollutants in large industrial plumes', *Atmos. Environ.* **42**, pp. 415-427, 2008.

Vione, D., Maurino, V., Minero, C., Pelizzitti, E., Harrison, M.A.J., Olariu, R-I. and Arsene, C., 'Photochemical reactions in the troposphere aqueous phase and on particulate matter', *Chem. Soc. Rev.*, 2006.

Redington, A.L. and Derwent, R.G., 'Calculation of sulphate and nitrate aerosol concentrations over Europe using a Lagrangian dispersion model', *Atmos. Environ.* **36**, pp. 4425-4439, 2002.

Ensembles

Galmarini, S., Bonnardot, F., Jones, A., Potemski, S., Robertson, L. and Martet, M., 'Multi-model vs. EPS-based ensemble atmospheric dispersion simulations: A quantitative assessment on the ETEX-1 tracer experiment case', *Atmos. Environ.* **44**, pp. 3558-3567, 2010.

Potemski, S. et al, 'Multi-model ensemble analysis of the ETEX-2 experiment', *Atmospheric Environment* **42**, pp. 7250-7265, 2008.

Becker, A., Wotawa, G., De Geer, L.-E., Seibert, P., Draxler, R., Sloan, C., D'Amours, R., Hort, M., Glaab, H., Heinrich, P., Grillon, Y., Shershakov, V., Katayama, K., Zhang, Y., Stewart, P., Hirtl, M., Jean, M. and Chen, P., 'Global backtracking of anthropogenic radionuclides by means of a receptor oriented ensemble dispersion modelling system in support of Nuclear-Test-Ban Treaty verification', *Atmos. Environ.* **41**, pp. 4520-4534, 2007.

Volcanic ash

Dacre, H.F., Grant, A.L.M., Harvey, N.J., Thomson, D.J., Webster, H.N. and Marenco, F., 'Volcanic ash layer depth: processes and mechanisms', *Geophys. Res. Lett.* **42**, pp. 637, 2015.

Schmidt, A., Witham, C.S., Theys, N., Richards, N.A.D., Thordarson, T., Szpek, K., Feng, W., Woolley, A.M., Johnson, B.T., Jones, A.R., Redington, A.L., Heard, I.P.C., Hort, M.C., Hayward, C.L. and Carslaw, K.S., 'Assessing hazards to aviation from sulfur dioxide emitted by explosive Icelandic eruptions', *J. Geophys. Res. Atmos.* **119**, pp. 14180-14196, doi:10.1002/asl2.506, 2014.

Turner, R., Moore, S., Pardo, N., Kereszturi, G., Uddstrom, M., Hurst, T. and Cronin, S., 'The use of Numerical Weather Prediction and a Lagrangian transport (NAME-III) and dispersion (ASHFALL) models to explain patterns of observed ash deposition and dispersion following the August 2012 Te Maari, New Zealand eruption', *J. Volc. and Geotherm. Res.* **286**, pp. 437-451, 2014.

Beckett, F.M., Witham, C.S., Hort, M.C., Stevenson, J.A., Bonadonna, C. and Millington, S.C., 'The sensitivity of NAME forecasts of the transport of volcanic ash clouds to the physical characteristics assigned to the particles', Met Office Forecasting Research Technical Report 592, Met Office, UK, 2014.

Devenish, B.J., Francis, P.N., Johnson, B.T., Sparks, R.S.J. and Thomson, D.J., 'Sensitivity analysis of dispersion modeling of volcanic ash from Eyjafjallajokull in May 2010', *J. Geophys. Res.* **117**, D00U21, 2012.

Heard, I.P.C., Manning, A.J., Haywood, J.M., Witham, C., Redington, A., Jones, A., Clarisse, L. and Bourassa, A., 'A comparison of atmospheric dispersion model predictions with observations of SO₂ and sulphate aerosol from volcanic eruptions', *J. Geophys. Res.* **117**, D00U22, 2012.

Webster, H.N., Thomson, D.J., Johnson, B.T., Heard, I.P.C., Turnbull, K., Marenco, F., Kristiansen, N.I., Dorsey, J., Minikin, A., Weinzierl, B., Schumann, U., Sparks, R.S.J., Loughlin, S.C., Hort, M.C., Leadbetter, S.J., Devenish, B.J., Manning, A.J., Witham, C.S., Haywood, J.M. and Golding, B.W., 'Operational prediction of ash concentrations in the distal volcanic cloud from the 2010 Eyjafjallajokull eruption', *J. Geophys. Res.* **117**, D00U08, 2012.

Witham, C., Webster, H., Hort, M., Jones, A. and Thomson, D.J., 'Modelling concentrations of volcanic ash encountered by aircraft in past eruptions', *Atmos. Environ.* **48**, pp. 219-229, 2011.

Animal health

Burgin, L.E., Gloster, J., Sanders, C., Mellor, P.S., Gubbins, S. and Carpenter, S., 'Investigating incursions of bluetongue virus using a model of long-distance Culicoides biting midge dispersal', *Transbound. Emerg. Dis.* **60**, Issue 3, pp. 263-272, 2013.

Gloster, J., Jones, A., Redington, A., Burgin, L., Sorensen, J.H., Turner, R., Dillon, M., Hullinger, P., Simpson, M., Astrup, P., Garner, G., Stewart, P., D'Amours, R., Sellers, R. and Paton, D., 'Airborne spread of foot-and-mouth disease - model intercomparison', *Vet. J.* **183**, pp. 278-286, 2010.

Chapman, J., Nesbitt, R., Burgin, L., Reynolds, D., Smith, A., Middleton, D. and Hill, J., 'Flight Orientation Behaviours Promote Longer Migration Trajectories in High-flying Insects', *Science* **327**, pp. 682-685, Supporting Online Material, 2010.

Sanders, C., Burgin, L., Pallot, A., Barber, J., Golding, N., Carpenter, S. and Gloster, J., 'A study of potential Bluetongue vectors and meteorology in Jersey', *Weather* **65(1)**, pp. 21-26, 2010.

Schley, D., Burgin, L. and Gloster, J., 'Predicting infection risk of airborne foot-and-mouth disease', *Atmospheric Environment* **42**, pp. 7250-7265, 2008.

Radiological

Draxler, R., Arnold, D., Chino, M., Galmarini, S., Hort, M., Jones, A., Leadbetter, S., Malo, A., Maurer, C., Rolph, G., Saito, K., Servranckx, R., Shimbori, T., Solazzo, E. and Wotawa, G., 'World Meteorological Organization's model simulations of the radionuclide dispersion and deposition from the Fukushima Daiichi nuclear power plant accident', *J. Environ. Rad.* **139**, pp. 172-184, 2015.

Leadbetter, S.J., Hort, M.C., Jones, A.R., Webster, H.N. and Draxler, R.R., 'Sensitivity of the modelled deposition of Caesium-137 from the Fukushima Dai-ichi nuclear power plant to the wet deposition parameterisation in NAME', *J. Environ. Rad.* **139**, pp. 200-211, 2015.

Bedwell, P., Wellings, J., Haywood, S.M., Jones, A.R. and Hort, M.C., 'Cloud Gamma Modelling in the UK Met Office's NAME III Model', In: AGU Chapman Conference on Advances in Lagrangian Modeling of the Atmosphere 1-1, 2011.

Bedwell, P., 'Implementation of decay chain modelling within NAME III', NAME III Documentation, MD4/1, 2009.

Bedwell, P., 'Implementation of cloud gamma modelling within NAME III', NAME III Documentation, MD4/2, 2009.

Emergency response applications

Leadbetter, S., Sibley, A. and Hort, M., 'Emergency planning and preparedness: The new CHEMET service from the Met Office', Chemical Hazards and Poisons Report 17, HPA, June 2010, http://www.hpa.org.uk/webc/HPAwebFile/HPAweb_C/1274091561553.

Buncefield oil depot fire

Webster, H.N., Devenish, B.J., Haywood, J.M., Lock, A.P. and Thomson, D.J., 'Using plume rise schemes to model highly buoyant plumes from large fires', In: *International Journal of Environment and Pollution*, Special Issue: Eleventh international conference on harmonisation within atmospheric dispersion modelling for regulatory purposes, Carruthers, D. and McHugh, C. (Eds.) pp. 226-234, 2011.

Devenish, B. J. and Edwards, J. M., 'Large-eddy simulation of the plume generated by the fire at the Buncefield oil depot in December 2005', *Proc. R. Soc. A*, doi:10.1098/rspa.2008.0288, 2008

Webster, H.N., Abel, S.J, Taylor, J.P., Thomson, D.J., Haywood, J.M. and Hort, M.C., 'Dispersion Modelling Studies of the Buncefield Oil Depot incident', Hadley Centre Technical Note No. 69, Met Office, 2006.

Inversion

Manning, A.J., O'Doherty, S., Jones, A.R., Simmonds, P.G. and Derwent, R.G., 'Estimating UK methane and nitrous oxide emissions from 1990 to 2007 using an inversion modeling approach', *J. Geophys. Res.* **116**, D02305, 2011.

Manning, A.J., Ryall, DB., Derwent R.G., Simmonds, P.G. and O'Doherty, S., 'Estimating European emissions of ozone-depleting and greenhouse gases using observations and a modelling back-attribution technique', *J. Geophysical Research* **108**, pp. 4405, 2003.

Dust and resuspended ash

Leadbetter, S.J., Hort, M.C., Lowis, von, S., Weber, K. and Witham, C.S., 'Modeling the resuspension of ash deposited during the eruption of Eyjafjallajökull in spring 2010', *J. Geophys. Res.* **117**, D00U10, 2012.

McConnell, C. L., Highwood, E. J., Coe, H., Formenti, P., Anderson, B., Osborne, S., Nava, S., Desboeufs, K., Chen, G. and Harrison, M. A. J., 'Seasonal variations of the physical and optical characteristics of Saharan dust: Results from the Dust Outflow and Deposition to the Ocean (DODO) experiment', *Journal of Geophysical Research* **113**, D14S05, doi:10.1029/2007JD009606, 2008.

Athanassiadou, M., Flocas, H., Harrison, M.A.J., Hort, M.C., Witham, C. and Millington, S., 'The dust event of 17 April 2005 over Athens, Greece', *Weather* **61**, pp. 125-131, 2006.

Observational flight campaigns

Webster, H.N., Witham, C.S., Hort, M.C., Jones, A.R. and Thomson, D.J., 'NAME modelling of aircraft encounters with volcanic ash plumes from historic eruptions', Met Office Forecasting Research Technical Report 552, Met Office, UK, 2013.

-
- Johnson, B., Turnbull, K., Brown, P., Burgess, R., Dorsey, J., Baran, A.J., Webster, H., Haywood, J., Cotton, R., Ulanowski, Z., Hesse, E., Woolley, A., and Rosenberg, P., 'In situ observations of volcanic ash clouds from the FAAM aircraft during the eruption of Eyjafjallajökull in 2010', *J. Geophys. Res.* **117**, D00U24, 2012.
- Chou, C., Formenti, P., Maille, M., Ausset, P., Helas, G., Harrison, M. and Osborne, S., 'Size distribution, shape, and composition of mineral dust aerosols collected during the African Monsoon Multidisciplinary Analysis Special Observation Period 0: Dust and Biomass-Burning Experiment field campaign in Niger, January 2006', *J. Geophys. Res.* **113**, D00C10, 2008.
- Johnson, B. T., Osborne, S. R., Haywood, J. M. and Harrison, M. A. J., 'Aircraft measurements of biomass burning aerosol over West Africa during DABEX', *Journal of Geophysical Research* **113**, D00C06, doi:10.1029/2007JD009451, 2008.

Bibliography

- Abdalmogith, S.S. and Harrison, R.M., 'The use of trajectory cluster analysis to examine the long-range transport of secondary inorganic aerosol in the UK', *Atmos. Environ.* **39**, pp. 6686-6695, 2005.
- Ackermann, I.J., Hass, H., Memmesheimer, M., Ziegenbein, C. and Ebel, A., 'The parameterization of the sulfate-nitrate-ammonia aerosol system in the long-range transport model EURAD', *Meteorol. Atmos. Phys.* **57**, pp. 101-114, 1995.
- Alfano, F., Bonadonna, C., Delmelle, P. and Costantini, L., 'Insights on tephra settling velocity from morphological observations', *J. Vol. Geotherm. Res.* **208**, pp. 86-98, 2011.
- Apsley, D.D., 'Modelling airborne dispersion of coarse particulate material', CEEB Report RD/L/3481/R89, 1989.
- Beckett, F.M., Witham, C.S., Hort, M.C., Stevenson, J.A., Bonadonna, C. and Millington, S.C., 'The sensitivity of NAME forecasts of the transport of volcanic ash clouds to the physical characteristics assigned to the particles', Met Office Forecasting Research Technical Report 592, Met Office, UK, 2014.
- Bedwell, P., Wellings, J., Haywood, S.M. and Hort, M.C., 'Cloud gamma modelling in the UK Met Office's NAME III model', in 13th Conference on Harmonisation within Atmospheric Dispersion Modelling for Regulatory Purposes, 1-4 June 2010, Paris, France, http://www.harmo.org/Conferences/Proceedings/_Paris/publishedSections/H13-017-abst.pdf, 2010.
- Collins, W.J., Stevenson, D.S., Johnson, C.E. and Derwent, R.G., 'Tropospheric ozone in a global scale three dimensional Lagrangian Model and its response to NO_x emission controls', *J. Atmos. Chem.* **26**, pp. 223-274, 1997.
- Dentener, F.J. and Crutzen, P.J., 'Reaction of N₂O₅ on tropospheric aerosols: impact on the global distributions of NO_x, O₃ and OH', *J. Geophys. Res.* **98**, pp. 7149-7163, 1993.
- Derwent, R.G., Jenkin, M.E., Utembe, S.R., Shallcross, D.E., Murrells, T.P. and Passant, N.R., 'Secondary organic aerosol formation from a large number of reactive man-made organic compounds', *Sci. Total Env.* **408 (16)**, p. 3374, 2010.

- Derwent, R.G. and Malcolm, A.L., 'Photochemical generation of secondary particulates in the United Kingdom', *Phil. Trans. R. Soc. Lond. A* **358**, pp. 1-15, 2000.
- Devenish, B.J., 'Using simple plume models to refine the source mass flux of volcanic eruptions according to atmospheric conditions', *J. Vol. Geotherm. Res.* **256**, pp. 118-127, 2013.
- Draxler, R., Arnold, D., Chino, M., Galmarini, S., Hort, M., Jones, A., Leadbetter, S., Malo, A., Maurer, C., Rolph, G., Saito, K., Servranckx, R., Shimbori, T., Solazzo, E. and Wotawa, G., 'World Meteorological Organization's model simulations of the radionuclide dispersion and deposition from the Fukushima Daiichi nuclear power plant accident', *J. Environ. Rad.* **139**, pp. 172-184, 2015.
- Emanuel, K.A., 'Atmospheric Convection', Oxford University Press, 1994.
- Fitzgerald, J.J., Brownell, G.L. and Mahoney, F.J., 'Mathematical theory of radiation dosimetry', Gordon and Breach Science Publishers, 1967.
- Ganser, G.H., 'A rational approach to drag prediction for spherical and nonspherical particles', *Powder Technology* **77**, pp. 143-152, 1993.
- Gardiner, C.W., 'Handbook of stochastic methods', Springer, 1985.
- de Haan, P. and Rotach, M.W., 'A novel approach to atmospheric dispersion modelling: the puff-particle model', *Q. J. R. Meteorol. Soc.* **124**, pp. 2771-2792, 1998.
- Harrison, R.M. and Kitto, A.M.N., 'Evidence for a surface source of atmospheric nitrous acid', *Atmos. Environ.* **28**, pp. 1089-1094, 1994.
- Hubbell, J.H., 'Photon mass attenuation and energy-absorption coefficients from 1 keV to 20 MeV', *Int. J. Appl. Radiat. Isot.* **33**, pp. 1269-1290, 1982.
- ICRP, 'Conversion coefficients for use in Radiological Protection against external radiation', ICRP Publication 74, Ann. ICRP, Vol. 26, No. 3/4, 1996.
- Jaeger, R.G. et al.(eds), 'Engineering compendium on radiation shielding. Vol. 1, Shielding fundamentals and methods', Springer-Verlag, 1968.
- Jones, A.R., 'Towards a probabilistic approach for hazard area assessment in dispersion modelling', in AGU Chapman Conference on Advances in Lagrangian Modeling of the Atmosphere, 9 – 14 October 2011, Grindelwald, Switzerland, http://www.empa.ch/plugin/template/empa/*/113606, 2011.
- Leadbetter, S.J., Hort, M.C., Jones, A.R., Webster, H.N. and Draxler, R.R., 'Sensitivity of the modelled deposition of Caesium-137 from the Fukushima Dai-ichi nuclear power plant to the wet deposition parameterisation in NAME', *J. Environ. Rad.* **139**, pp. 200-211, 2015.

- Malcolm, A.L., Derwent, R.G. and Maryon, R.H., 'Modelling the long range transport of secondary PM10 to the UK', *Atmos. Environ.* **34**, pp. 881-894, 2000.
- Maryon, R.H., 'The gravitational settling of particulates: towards a parametrization for the NAME dispersion model', Turbulence and Diffusion Note 244, Met Office, UK, 1997.
- Maryon, R.H., Ryall, D.B. and Malcolm, A.L., 'The NAME 4 Dispersion Model: Science Documentation', Turbulence and Diffusion Note 262, Met Office, UK, 1999.
- Mastin, L.G., Guffanti, M., Servranckx, R., Webley, P., Barsotti, S., Dean, K., Durant, A., Ewert, J.W., Neri, A., Rose, W.I., Schneider, D., Siebert, L., Stunder, B., Swanson, G., Tupper, A., Volentik, A. and Waythomas, C.F., 'A multidisciplinary effort to assign realistic source parameters to models of volcanic ash-cloud transport and dispersion during eruptions', *J. Vol. Geotherm. Res.* **186**, pp. 10-21, 2009.
- Mastin, L.G., 'Testing the accuracy of a 1-D volcanic plume model in estimating mass eruption rate', *J. Geophys. Res.* **119**, pp. 2474-2495, 2014.
- Meneguz, E. and Thomson, D.J., 'Towards a new scheme for parametrisation of deep convection in NAME III', *Int. J. Environment and Pollution* **54**, pp. 128-136, 2014.
- Meng, Z. and Seinfeld, J.H., 'Time scales to achieve atmospheric gas-aerosol equilibrium for volatile species', *Atmos. Environ.* **30**, pp. 2889-2900, 1996.
- Morrison, N.L. and Webster, H.N., 'An assessment of turbulence profiles in rural and urban environments using local measurements and numerical weather prediction results', *Boundary-Layer Meteorol.* **115**, pp. 223-239, 2005.
- Mueller, E.H., Ford, R., Hort, M.C., Huggett, L., Riley, G. and Thomson, D.J., 'Parallelisation of the Lagrangian atmospheric dispersion model NAME', *Comp. Phys. Comms.* **184**, pp. 2734-2745, 2013.
- Pal Arya, S., 'Air pollution meteorology and dispersion', Oxford University Press, 1999.
- Pasquill, F. and Smith, F.B., 'Atmospheric diffusion', Ellis Horwood, Chichester, 1983.
- Pope, S.B., 'Turbulent flows', Cambridge University Press, 2000.
- Pruppacher, H.R. and Klett, J.D., 'Microphysics of Clouds and Precipitation', Kluwer, Dordrecht, 1997.
- Raza, S. and Avila, R., 'A 3D Lagrangian particle model for direct plume gamma dose rate calculations', *J. Rad. Prot.* **21** (2), pp. 145-154, 2001.
- Redington, A.L., Derwent, R.G., Ryall, D.B., Matthew, S. and Manning, A.J., 'Pollution of the Urban Midlands Atmosphere: Development of an "urban airshed" model

- for the West Midlands', Hadley Centre Technical Note 31, Met Office, Bracknell, <http://www.metoffice.gov.uk/research/hadleycentre/pubs/HCTN/index.html>, 2001.
- Redington, A.L. and Derwent, R.G., 'Calculation of sulphate and nitrate aerosol concentrations over Europe using a Lagrangian dispersion model', *Atmos. Environ.* **36**, pp. 4425-4439, 2002.
- Riley, C.M., Rose, W.I. and Bluth, G.J.S., 'Quantitative shape measurements of distal volcanic ash', *J. Geophys. Res.* **108**, pp. B10,2504, 2003.
- Rodean, H.C., 'Stochastic Lagrangian models of turbulent diffusion', American Meteorological Society, 1996.
- Schmidt, A., Witham, C.S., Theys, N., Richards, N.A.D., Thordarson, T., Szpek, K., Feng, W., Woolley, A.M., Johnson, B.T., Jones, A.R., Redington, A.L., Heard, I.P.C., Hort, M.C., Hayward, C.L. and Carslaw, K.S., 'Assessing hazards to aviation from sulfur dioxide emitted by explosive Icelandic eruptions', *J. Geophys. Res. Atmos.* **?**, to appear, 2015.
- Simmonds, J.R., Lawson, G. and Mayall, A., 'Methodology for assessing the radiological consequences of routine releases of radionuclides to the environment', RP 72, EUR 15760 EN, 1995.
- Sparks, R.S.J., Bursik, M.I., Carey, S.N., Gilbert, J.S., Glaze, L.S., Sigurdsson, H. and Woods, A.W., 'Volcanic plumes', John Wiley & Sons, 1997.
- Staniforth, A. and Côté, J., 'Semi-Lagrangian integration schemes for atmospheric models – a review', *Monthly Weather Review* **119**, pp. 2206-2223, 1991.
- Stevenson, J.A., Loughlin, S.C., Font, A., Fuller, G.W., MacLeod, A., Oliver, I.W., Jackson, B., Horwell, C.J., Thordarson, T. and Dawson, I., 'UK monitoring and deposition of tephra from the May 2011 eruption of Grímsvötn, Iceland', *J. Appl. Volcanology* **2**, pp. 3, 2013.
- Taylor, G.I., 'Diffusion by continuous movements', *Proc. Lond. Math. Soc.* **2** **20**, pp. 196-212, 1922.
- Thomson, D.J., 'Criteria for the selection of stochastic models of particle trajectories in turbulent flows', *J. Fluid Mech.* **180**, pp. 529-556, 1987.
- Thomson, D.J., 'The fluctuations module', ADMS Technical Specification paper P13/01E/92, CERC, 1992.
- Thomson, D.J., 'Averaging time and fluctuations in ADMS versions 1 and 2', ADMS Technical Specification paper P13/03C/96, CERC, 1996.
- Thomson, D.J., 'Concentration fluctuations in ADMS 3, including fluctuations from anisotropic and multiple sources', ADMS Technical Specification paper P13/07D/00, CERC, 2000.
- Thomson, D.J., 'The met input module', ADMS Technical Specification paper P05/01R/12, CERC, 2012.

- Thomson, D.J. and Jones, A.R., 'A new puff modelling technique for short range dispersion applications', *Int. J. Environment and Pollution* **44**, pp. 156-163, 2011.
- Thomson, D.J. and Wilson, J.D., 'History of Lagrangian stochastic models for turbulent dispersion', In: *Lagrangian modelling of the atmosphere*, Geophysical Monograph Series 200, American Geophysical Union, Washington, pp. 19-36, 2013.
- Turner, R., Moore, S., Pardo, N., Kereszturi, G., Uddstrom, M., Hurst, T. and Cronin, S., 'The use of Numerical Weather Prediction and a Lagrangian transport (NAME-III) and dispersion (ASHFALL) models to explain patterns of observed ash deposition and dispersion following the August 2012 Te Maari, New Zealand eruption', *J. Volc. and Geotherm. Res.* **286**, pp. 437-451, 2014.
- Underwood, B.Y., 'A review of dry and wet deposition', AEAT-5382, AEA Technology plc, UK, 1999.
- Webster, H.N., Thomson, D.J. and Morrison, N.L., 'New turbulence profiles for NAME', Turbulence and Diffusion Note 288, Met Office, UK, 2003.
- Webster, H.N. and Thomson, D.J., 'Validation of a Lagrangian model plume rise scheme using the Kincaid data set', *Atmos. Environ.* **36**, pp. 5031-5042, 2002.
- Webster, H.N. and Thomson, D.J., 'Parameterising low-frequency meander in atmospheric dispersion models', In: Tenth international conference on harmonisation within atmospheric dispersion modelling for regulatory purposes, Skouloudis, A.N., Kassomenos, P. and Bartzis, J. (Eds.), pp. 594-598, 2005.
- Webster, H.N. and Thomson, D.J., 'Dry deposition modelling in a Lagrangian dispersion model', *Int. J. Env. Pollution* **47**, pp. 1-9, 2011.
- Webster, H.N. and Thomson, D.J., 'The NAME wet deposition scheme', Met Office Forecasting Research Technical Report 584, Met Office, UK, 2014.
- Webster, H.N. and Thomson, D.J., 'Parametrizing unresolved mesoscale motions in NAME', Met Office Forecasting Research Technical Report 601, Met Office, UK, 2015.
- White, F.M., 'Viscous Fluid Flow', McGraw-Hill, 1974.
- Wilson, L. and Huang, T.C., 'The influence of shape on the atmospheric settling velocity of volcanic ash particles', *EPSL.* **44**, pp. 311-324, 1979.
- Woodhouse, M.J., Hogg, A.J., Phillips, J.C. and Sparks, R.S.J., 'Interaction between volcanic plumes and wind during the 2010 Eyjafjallajökull eruption, Iceland', *J. Geophys. Res.* **118**, pp. 92-109, 2013.
- Wright, B.J., 'A variational rainfall rate analysis for Nimrod', Met Office Forecasting Research Technical Report 143, Met Office, UK, 1994.

Index

- absolute times, 14
- accuracy of coordinate conversions, 19
- activity, 122
- advanced flow options, 92
- advection and diffusion, 100
- agent decay, 62, 126
- air history maps, 28
- air kerma, 124
- air quality, 8, 26
- animal health, 8
- array structures, 16
- atmospheric stability, 104
- atmospheric turbulence, *see* turbulence
- averaging over time, *see* temporal averaging
- Bateman equations, 123
- becquerel Bq, 122
- below-cloud scavenging, *see* washout
- bespoke decay schemes, 127
- bibliographical references, 167
- biogenic α -pinene emissions, 70
- Bluetongue, 8
- boundary layer entrainment, 108
- bounded domain, 25
- branching ratio, 61, 123
- Brownian motion, 101
- building flow, *see* flow deformation effects of an isolated building
- Buncefield oil depot fire, 78
- buoyant releases, 70, 110
- calculating boundary layer averages, 45
- calculating maximum value over ensemble, 49
- calculating maximum value over time, 48
- calculating median value over ensemble, 49
- calculating median value over time, 48
- calculating minimum value over ensemble, 49
- calculating minimum value over time, 48
- calculating statistics for concentration fluctuations, 50, 135
- calculating statistics over an ensemble, 48
- calculating statistics over time, 47
- calculating time average or integral, *see* temporal averaging
- calculating vertical integrals, 45
- calling LINCOM from within NAME, 92
- cannonballs* and *snowflakes*, 74
- case, *see* running multiple cases
- Case 'C', 49
- Chalara ash dieback, 8
- CHEMET, 8
- chemistry, 127
- clipped Gaussian distribution, 135
- cloud gamma dose outputs, 125
- cloud gamma doses, 61, 63, 124
- cloud gamma parameters, 63, 126
- cloud shine, *see* cloud gamma doses
- code profiling, 144
- code structure, 137
- Combined quantities, 39, 43
- command line arguments, 35, 140
- comments in NAME input files, 36
- compilation, 137
- compilation script, 138
- compiler directives, 137, 139

- compiler options, 139, 144
- compiling on Linux systems, 138
- compiling on Windows systems, 139
- computational domain, 24
- concentration fluctuations, 50, 134
- constraining lifetime of particles, 25
- contiguous vertical layers, 22
- continuous releases, 66
- contributions to time-averaged quantities, 46
- controlling deep convection, 110
- controlling particle numbers in a run, 73, 96
- controlling sampling for statistics, 47
- controlling the turbulence and mesoscale motions
 - parametrizations, 107
- controlling warnings and errors, 32
- convective mixing, *see* vertical mixing by deep convection
- conventions, 6
- conversion of output format, *see* post-processing of NAME outputs
- Copyright notice, 2
- cores, 141
- correspondence between species and particles, 63
- creating graphical products, 154
- creating groups of sources, *see* source groups
- critical threshold for precipitation, 117
- CTBTO, 8
- Cunningham slip correction, 120
- damped eddy diffusivity, 101, 108
- daughter products, 61, 123
- decay chains, *see* radiological decay chains
- defining a seasonal variation for emissions, 67
- defining a traffic cycle for vehicle emissions, 67
- defining source terms, *see* sources
- defining species information, *see* species
- defining species use information, *see* species uses
- depletion equation, 116
- deposition outputs, 115, 117
- deposition velocity, 61, 112
- diffusive scheme, 101
- discrete vertical levels, 22
- disk output, 37
- dispersion options, *see* specifying dispersion model options
- documentation, *see* model documentation
- domain-integrated quantities, 45
- domains, 24
- dose rate, 124
- drag coefficient of a particle, 119
- dry deposition, 61, 111, 134
- dust source, 18
- dynamic and convective precipitation, 116
- ECMWF Ensemble Prediction System, 86
- ECMWF GRIB API, 149
- eddy diffusivity, 101
- EMEP Grid, 19
- ensemble average, *see* ensemble mean
- ensemble mean, 48, 87
- ensemble median, 87
- equation of motion, 100
- error file, 32
- Eulerian model, 128, 132
- Eulerian quantities, 39, 43
- European Centre for Medium-Range Weather Forecasts (ECMWF), 78, 149
- exceedence probabilities, *see* probabilities
- explicit spatial averaging, 45
- Eyjafjallajökull eruption 2010, 8
- fatal error, 32
- FCM, 137
- field column headers, 52
- field-based outputs, *see* output fields
- filename conventions, 35

- finite-duration releases, 66
- fixed met option, 29
- flight levels, 20
- flow attributes, 83
- flow deformation effects of an isolated building, 75, 93
- flow domain, 24
- flow module, 75
- Flow quantities, 40, 43
- Foot-and-Mouth Disease, 8
- formatting of output files, 49, 51
- Fortran 90 source code, 137
- free-tropospheric turbulence, 106
- full* particles, 96
- Ganser sedimentation scheme, 119
- gaseous species, 113
- Gaussian source distribution, 65
- general vertical layers, 22
- geographical locations, 21, 65
- global control of particle numbers, 74
- GNU profiler, 144
- graphical output, 37
- gravitational settling, *see* sedimentation
- GRIB, 147, 149, 157
- gridded met-dependent source, 68
- gridded meteorological fields, *see* NWP meteorology
- gridded source, *see* gridded met-dependent source
- Grimsvotn eruption 2011, 8
- guidance on compilation using the Intel compiler, 138
- half life, 60, 122
- handling of folder delimiters, 35
- headed input files, *see* input files
- height above ground level, 20
- height above mean sea level, 20
- homogeneous turbulence profiles, 104
- horizontal coordinate systems, 19
- horizontal grids, 21
- hybrid coordinate systems (eta coordinates), 20
- ICAO standard atmosphere, 14, 20
- IDL visualisation software, 154
- in-cloud scavenging, *see* rainout
- indexed grid, *see* discrete vertical levels
- infinite times, 15
- informational messages, 32
- inhomogeneous turbulence profiles, 104
- initialising from a 'spun-up' state, 26
- input files, 35
- input meteorology, *see* meteorological data
- installation and model upgrades, 164
- instantaneous releases, 66
- integrating over time, *see* temporal averaging
- Intel Visual Fortran Studio, 139
- interactive use, 140
- interface levels, *see* contiguous vertical layers
- interpretation of gridded values, 22, 23, 45
- interpreting times in back runs, 29
- introducing the random-walk schemes, 100
- inversion modelling, 8
- invoking the chemistry scheme, 127
- iodine chemistry, 131
- Iris, 156
- irregularly-spaced grids, 21
- iterative plume-rise model, 69
- killing model particles, 25
- Lagrangian approach, 100
- Lagrangian particle versus semi-infinite cloud approach, 124
- Lagrangian quantities, 39, 42
- Lagrangian timescales, 101
- land-use dependent dry deposition, 61, 83, 114
- land-use fractions, 114

- Langevin equation, 101
- Langevin-type approach, 103
- latitude-longitude coordinates, 19
- Legionnaire's disease, 8
- level of agreement, 87
- licensing of NAME, 163
- LINCOM, 92
- linear flow model, 92
- list of trajectory parameters, 56
- locations, *see* geographical locations
- log file, 32
- long-range random-walk scheme, *see* diffusive scheme
- loss processes, 111

- Mace Head, 8
- makefile, 138, 145
- managing NWP data files, 81
- mass release rate, *see* specifying mass release for a source
- material units, 60
- maximum deposition height, 114
- maximum number of particles in a run, 67, 94
- maximum number of puffs in a run, 98
- mean aerosol diameter, 61, 113
- meander, *see* unresolved mesoscale motions
- message controls, *see* controlling warnings and errors
- met ensemble size, *see* number of ensemble members
- met module, 75
- met-dependent source for volcanic eruptions, 69
- met-dependent sources, 70
- meteorological data, 75
- meteorological uncertainty, 86
- meteorological utilities, 147
- midges, 70
- mineral dust, 68, 83
- miscellaneous species characteristics, 62
- model accuracy versus speed, 73
- model documentation, 159
- modelling near-source dispersion, 102, 115
- modelling sedimenting material, 71
- modelling stack releases, 110
- modifying turbulence parameters, 78, 82
- MOGREPS, 86
- moist convection, 109
- molecular weight, 62
- momentum-driven releases, 70, 110
- multi-core processors, 141
- multi-model ensemble, 88
- multiple dispersion modelling options, 29, 94

- NAME
 - advanced modelling options, 26
 - applications, 8
 - back runs, 28
 - bibliography, 167
 - calendar times, 14
 - chemistry, 127
 - code management, 137
 - compiling the source code, 137
 - core model components, 14
 - deposition and sedimentation, 111
 - documentation, 159
 - Eulerian model, 132
 - external interfaces, 8, 140
 - external users, 8
 - inputs, 35
 - installation, 164
 - introduction, 7
 - Lagrangian approach, 9
 - licensing, 163
 - meteorology, 75
 - model overview, 11
 - NAME II, 6, 54

- non-radiological decay processes, 126
- outputs, 37
- parallelisation, 141
- physical processes, 94
- post-processing of outputs, 157
- producing graphics, 154
- profiling, 144
- profiling and use of timers, 144
- radiological modelling, 121
- random-walk schemes, 100
- representation of times, 14
- running the model, 140
- sources, 59
- supplementary utilities and programs, 147
- supported operating systems, 7
- testing and validation, 162
- time intervals, 14
- training, 165
- user community, 166
- using particles, 94
- using puffs, 97
- name2adms file-conversion utility, 148
- near-source* scheme, 96, 104, 107, 111
- nested meteorological data, 75
- NetCDF, 147, 152, 157
- NetCDF API, 152
- nominal time of a puff, 98
- non-fatal error, 32
- non-repeatable runs, 28
- non-sedimenting particles, 62, 113
- non-spherical particles, 119
- number of ensemble members, 88
- Numerical Weather Prediction (NWP), 75, 78
- NWP met extraction run, 148
- NWP meteorology, 75, 78
 - ensembles, *see* using ensemble NWP meteorology
 - how to set up, 79
 - how to set up ancillaries, 84
 - loading met data on demand, *see* using update-on-demand
 - model orography and other ancillaries, 79
 - organisation of ensemble met data, 88
 - restore met option, *see* using a met restore script
 - setting limits on the boundary layer depth, 81
 - the met definition file, 79
 - use of NWP data sets, 78
- Open Multi-Processing, *see* use of OpenMP
- optimising NAME applications, 144
- original puffs, 98
- Ornstein-Uhlenbeck process, 102
- Other field quantities, 41, 44
- output fields, 37, 52
- output files at a restart, 54
- output groups, 52
- overview of chemistry scheme, 129
- parallel computing, 141
- parallel met read, *see* using a dedicated I/O thread
- parallelisation, 31, 142
- particle mass limits, 66, 71, 74
- particle release rate, *see* specifying number of particles for a source
- particle size distributions, 16, 68, 71
- particle splitting, 71
- particle trajectories, *see* particle/puff information
- particle/puff information, 49, 56
- percentiles, 47, 48, 50, 87
- photon energy, 64, 124
- physical constants, 14
- plant ancillaries, 83, 114
- plotting and analysis procedures for NAME outputs, 154
- plume rise, 70, 96, 110

- Polar Stereographic projection, 19
- poor-man's ensemble, 88
- post-processing of NAME outputs, 157
- power law decay, 63, 126
- pre-built executables, 138
- Preface, 1
- pressure as a vertical coordinate, 20
- probabilistic forecasting, *see* meteorological uncertainty
- probabilities, 47, 48, 50, 87
- probability distribution of short-term concentration, 135
- producing ADMS-style met files, 148
- producing graphics using IDL, 154
- producing graphics using Python, 156
- profiling tools, 144
- programming standards, 137
- prototype* meteorology, 93
- puff interval, 99
- puff model, 97
- puff splitting and recombination, 98
- puff time, 98
- puff-to-particle transition time, *see* puff time
- Python, 156
- radar met
 - definition file, 89
 - how to set up, 90
 - restore met option, *see* using a radar met restore script
- radar rainfall, 89
- radioactive decay, 60, 122
- radioactive decay of deposited material, 115, 118, 122
- radiological characteristics, 60
- radiological decay chains, 61, 123
- rainout, 62, 115
- random number generator, 27
- random seed, *see* random number generator
- reaching the global particle limit, 74, 96
- reading GRIB met data files, 149
- reading NetCDF met data files, 152
- receptor-oriented modelling, *see* running NAME backwards
- recycling of particles, 96
- region of validity, *see* domains
- relative times, 14
- releasing particles of one particular size, 71
- releasing particles over a particle size range, 71
- repeatable runs, 27
- reproducibility of NAME results, 143
- requesting deposition outputs, *see* deposition outputs
- requesting output of fields, 37
- requesting output of particle trajectories, 49
- requesting output of pdfs for concentration fluctuations, 50, 135
- requesting outputs for multiple species, 38
- requesting timing information, 146
- resistance analogy for deposition velocity, 112
- restart files, 27
- restart functionality, 26
- restricting output to a particular source, 38
- restricting output to a source group, 72
- restricting the mass on individual particles, 71
- resuspended volcanic ash, 70
- RIMNET, 8
- Risø, 92
- RSMC, 8
- running multiple cases, 29, 48, 87, 94
- running NAME, 140
- running NAME backwards, 28
- running out of particles, *see* reaching the global particle limit
- running via a script or batch file, 140

-
- sampling period and rate for statistics, 47
 - scavenging coefficient, 62, 116
 - scavenging parameters (recommended values of), 117
 - screen output, 37
 - sea salt, 68
 - sea-salt source, 17
 - sedimentation, 70, 118, 133
 - sedimentation velocity, 118
 - sedimenting particles, 62, 113, 118
 - sedimenting releases, 70, 118
 - self-guide tutorial, 165
 - semi-Lagrangian advection, 132
 - setting system or environment variables, 140, 151
 - shared memory, 141
 - short-range random-walk scheme, *see* velocity-memory scheme
 - single-site meteorology, 75, 76
 - differences between NAME and ADMS, 76, 148
 - how to set up, 77
 - met pre-processor, 76, 149
 - specifying a time-zone, 77
 - skewed turbulence in convective conditions, 105, 108
 - small travel-time behaviour, *see* damped eddy diffusivity
 - small-scale terrain effects, 75
 - soil ancillaries, 83
 - source attribution, 72
 - source geometry, 65
 - source groups, 72
 - source location, 65
 - source mass flux for volcanic eruptions, 69
 - source time dependency, 67
 - sources, 64
 - spatial averaging, *see* explicit spatial averaging
 - special treatment of deposition for ozone and hydrogen, 113
 - species, 59
 - species uses, 63
 - specifying a flow domain, 78, 82, 91
 - specifying a maximum deposition height, *see* maximum deposition height
 - specifying a source size in metres, 65
 - specifying chemistry options, 128
 - specifying dispersion model options, 94
 - specifying grids for chemistry and Eulerian model, 128
 - specifying mass release for a source, 66
 - specifying number of particles for a source, 66, 73
 - specifying parallelisation options, 142
 - specifying percentile values, 48
 - specifying receptor sites, 21
 - specifying soil characteristics, *see* soil ancillaries
 - specifying source locations, 21
 - specifying surface characteristics, *see* surface ancillaries
 - specifying the synchronisation time, 46, 129
 - specifying threshold values for probabilities, 48
 - specifying trajectory options, 49
 - specifying vegetation characteristics, *see* plant ancillaries
 - spherical particles, 119
 - splitting trajectory information in separate files, 57
 - stable species, 60
 - standard deviation of short-term concentration, 135
 - standard format for horizontal fields, 54
 - standard format for time series, 54
 - statistical noise, 9, 97
 - statistical processing, 47, 86
 - STOCHEM, 114, 128

- structured horizontal grids, 21
- style syntax, 6
- supplementary qualifiers for a field request, 38
- surface ancillaries, 83, 114
- surface resistance, 61, 112
- suspending a run, 30
- Sync? keyword, 115, 118, 129
- temporal averaging, 45
- temporal extent of a release, 66
- temporal grids, 23
- testing and validation, 162
- the 'run-to' file, 30
- threads, 142
- time grids, *see* temporal grids
- time zones, 15
- time-lagged ensemble, 88
- time-spread of a puff, 98
- time-varying sources, 67
- timers, 143, 145
- timing log file, 146
- top hat* distribution, *see* uniform source distribution
- trace gas measurements, 8
- training course, 165
- trajectory-based outputs, *see* particle/puff information
- trajectory-crossing and inertial effects for settling particles, 106, 121
- Transverse Mercator projection, 19
- treating multiple decay-chain pathways, 123
- treatment of radionuclides, *see* radioactive decay
- treatment of zero values in field output, 54
- tropospheric chemistry, 128
- turbulence, 100, 104
- UK National Grid, 19
- unbounded domain, 25
- uncertainties in dispersion modelling, 86
- understanding the performance of NAME, 144
- unexpected fatal error, 32
- Unified Model, 10, 78, 83
- uniform source distribution, 65
- uniform vertical grid, 22
- unit conversions, 60
- units, 14
- unresolved mesoscale motions, 100, 106
- unstructured horizontal grids, 21
- update-on-demand, 28
- urban scheme, 83
- use of local tables by the GRIB API, 150, 151
- use of OpenMP, 31, 141
- use of timers, *see* timers
- user community, 166
- using a dedicated I/O thread, 32, 143
- using a met restore script, 81
- using a radar met restore script, 90
- using a resistance-based deposition velocity, 113
- using an explicit deposition velocity, 112
- using ensemble NWP meteorology, 29, 86
- using multiple input files, 35
- using multiple sets of input meteorology, 75, 79
- using multiple threads, *see* parallelisation
- using NWP ancillaries, 68, 83
- using rainfall analyses, *see* radar rainfall
- using the GRIB API, 150
- using the NetCDF API, 152
- using the random-walk schemes, 107
- using update-on-demand, 81, 91
- using weather station observations, *see* single-site meteorology
- UTC (Universal Coordinated Time), 15
- UV decay, *see* agent decay
- UV loss rate, 62, 127
- validation studies, 159, 162

velocity memory, 101

vertical coordinate systems, 20

vertical diffusion, 133

vertical grids, 22

vertical mixing by deep convection, 109

Volcanic Ash Advisory Centre (VAAC), 8

VTune, 144

warning, 32

washout, 62, 115

well-mixed condition, 103

wet deposition, 62, 115, 134

White sedimentation scheme, 119

Wiener process, 101

Wilson and Huang sedimentation scheme, 120

Met Office

FitzRoy Road, Exeter
Devon, EX1 3PB
UK

Tel: 0370 900 0100

Fax: 0370 900 5050

enquiries@metoffice.gov.uk

www.metoffice.gov.uk