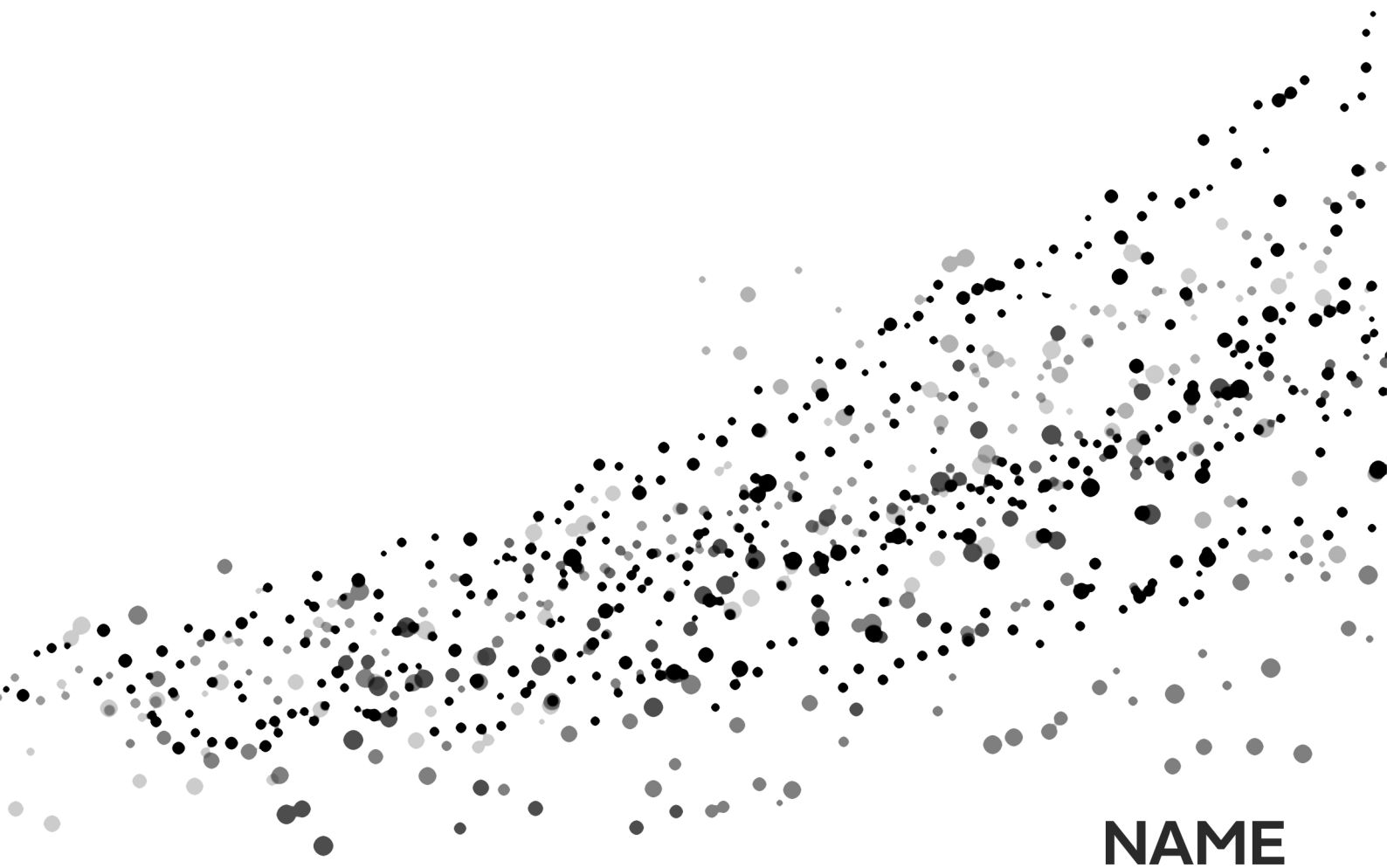
Met Office

NAME Technical Specification Document A02

NAME Installation and Troubleshooting Guide

Andrew Jones and Eike Mueller

Documentation issued with	**:**	NAME Version 8.7
Document last updated for	**:**	NAME Version 8.7
Last updated on	**:**	16/04/2025



NAME

Numerical Atmospheric-Dispersion Modelling Environment

Met Office, FitzRoy Road, Exeter, Devon, EX1 3PB, United Kingdom

Contents

1 Introduction

This document describes the installation procedure for the NAME model, including instructions for testing your NAME installation. It also discusses some of the problems that might be encountered when the model is run on your particular system. The troubleshooting guide suggests solutions for some of the more common issues that have been experienced by external users when setting up NAME for the first time.

2 Supported platforms

The NAME dispersion model is written in the Fortran 90 programming language, though is increasingly taking advantage of advanced features in more recent Fortran standards available in modern compilers. NAME has an MPI implementation from version 8.0 onwards, which requires an appropriate MPI environment to be available when compiling and running the model – more details are provided on the MOSRS page *"Compiling, Running and Debugging NAME"*. The Met Office has built, tested and supports NAME using the Intel Fortran compiler running under the RedHat Linux, Cray Linux and Microsoft Windows operating systems[1]. Version 17.0 or later of the Intel compiler is currently recommended. Pre-built executable files are sometimes provided as part of a NAME distribution. These executables are appropriate for many NAME users and their use is generally recommended. Use of a pre-built executable also simplifies the installation process and avoids the requirement for a Fortran compiler to be available on the installation platform.

When appropriate, a NAME executable can be compiled from its source code. For example, this would be necessary when using a different operating system to the supported operating systems listed above, when a different compiler is being used, or when changes have been made to the source code (e.g. by research users of the model). In principle, NAME should be able to run on any platform for which a modern Fortran compiler supporting MPI and OpenMP is available. Any external users intending on compiling the NAME source code can seek further guidance. To date, NAME has also been compiled and run under the following platforms and compilers:

- Sun Solaris O/S
- AIX on IBM HPC
- Cray compiler on Cray HPC
- gfortran (GNU Fortran) on Linux
- Apple Macintosh

3 System requirements

System requirements for the installation of the NAME model are minimal in the context of modern computer systems. The main distribution of NAME III requires about 100 MB of free space on the hard disk. It is recommended to copy the full distribution, however a more selective installation of folders is possible if local disk space is an issue.

System requirements for running NAME are largely determined by the modelling application being considered. NAME can be run in a wide variety of configurations, which influence the load placed on the host computer. An illustration of typical computational demands for various modelling configurations is presented in Table 1. Model runs using NWP met data can require significantly more computing resources than when using simple single-site met data. Guidance on the size of the available NWP data sets is provided in separate documentation. Resources are also determined by factors such as the number of particles in the simulation, the size of the geographical domain and duration of the simulation period being considered, the number of output requests, and the use of more complex modelling options such as chemistry.

Note that the amount of RAM that a single application can address in 32-bit operating systems is limited[2]. In theory the limit is 3 GB, but in practice we have found the limit to be just under 2 GB on both Linux and Windows

[1] For users on Windows, further instructions and configuration files for building NAME with Visual Studio are provided in the folder `VisualStudioProjects`.

[2] Of course, there is also a theoretical limit on the memory that can be addressed on 64-bit operating systems, although in practice this is much greater than the total virtual RAM available on current systems.

Application	*Typical met data*	*Typical RAM usage (GB)*
Short-range emergency response (1 km – 200 km, single source)	Local observations or high resolution NWP data	0.3 - 2.0
Long-range emergency response (100's km – 1000's km, single source)	Global NWP data	0.7 - 4.0
Air quality (multiple species/sources with modelling of atmospheric chemistry)	Global NWP data (European region)	2.0 - 6.0

Table 1: Some typical configurations of the NAME model

systems. The recommendation is therefore to limit NAME applications to less than 2 GB on 32-bit operating systems, and to use 64-bit systems whenever applications are likely to exceed this limit.

Model run time is determined by the demands of the simulation, but is also dependent on other factors such as parallelisation options, the CPU speed and other hardware influences. NAME run times can be as short as a few tens of seconds but can extend to several months of computing time. NAME has been run over a broad spectrum of computing systems, from a lowly laptop to a state-of-the-art high-end Linux server or supercomputing platform. As a general rule, a more powerful system will enable larger and faster NAME runs to be performed.

Provided that there is enough RAM available on a system, then it is quite possible to launch multiple NAME runs on the same computer. Our general advice is that the total number of threads allocated for NAME runs (including parallelised NAME runs) should not normally exceed the number of available cores. We have generally found that the optimum number of threads for NAME is influenced by the type of modelling application and also varies from system to system. It can only really be determined for a particular configuration through experimenting with the set up.

4 NAME installation

The installation instructions given here assume that a pre-built executable file is being used. Further advice should be sought from us in instances where the source code is being compiled.

Installation of the NAME model is a simple and easy procedure: **the NAME distribution downloaded from the Met Office Science Repository Service (or supplied on an installation DVD) should be copied to a folder on the host computer**. There is no installation script or wizard, and no requirement for the computer to be restarted after installation. Figure 1 shows the folder structure of the NAME distribution.

Although there is no reason why the installation folder cannot be used to store the local files that you will create, it is generally recommended to set up a separate folders structure to store local files. This should help to ensure that the NAME distribution is not accidentally corrupted. Your local files might include NAME input files (including sources files), met data and output files from your NAME runs.

Use of optional third-party software packages: If NAME is to be used with packed PP, GRIB or netCDF format files, some additional third-party software is required and, if not already available on your system, will need to be installed; see the separate guidance on this in the top-level `ReadMe.txt` file.

Although it is not essential to set up a local folder structure at the time of installation, it is a good time to consider how your local data might be organised. One approach would be to follow a similar folder structure as used by the NAME model, or you may prefer to use your own layout. NAME can be very flexible in how it references its input and output files in whatever approach you decide to adopt. Items to consider here include how to arrange your input files for NAME runs, where to write your output files, and how to store your met data files.

5 Testing your NAME installation

The NAME model is run by launching its executable file. Prior to v8.0, the executable could be run directly by simply calling it at the command line or by invoking it within a script. Following the implementation of MPI in NAME at v8.0, running the model has become slightly more involved and requires the executable to be launched

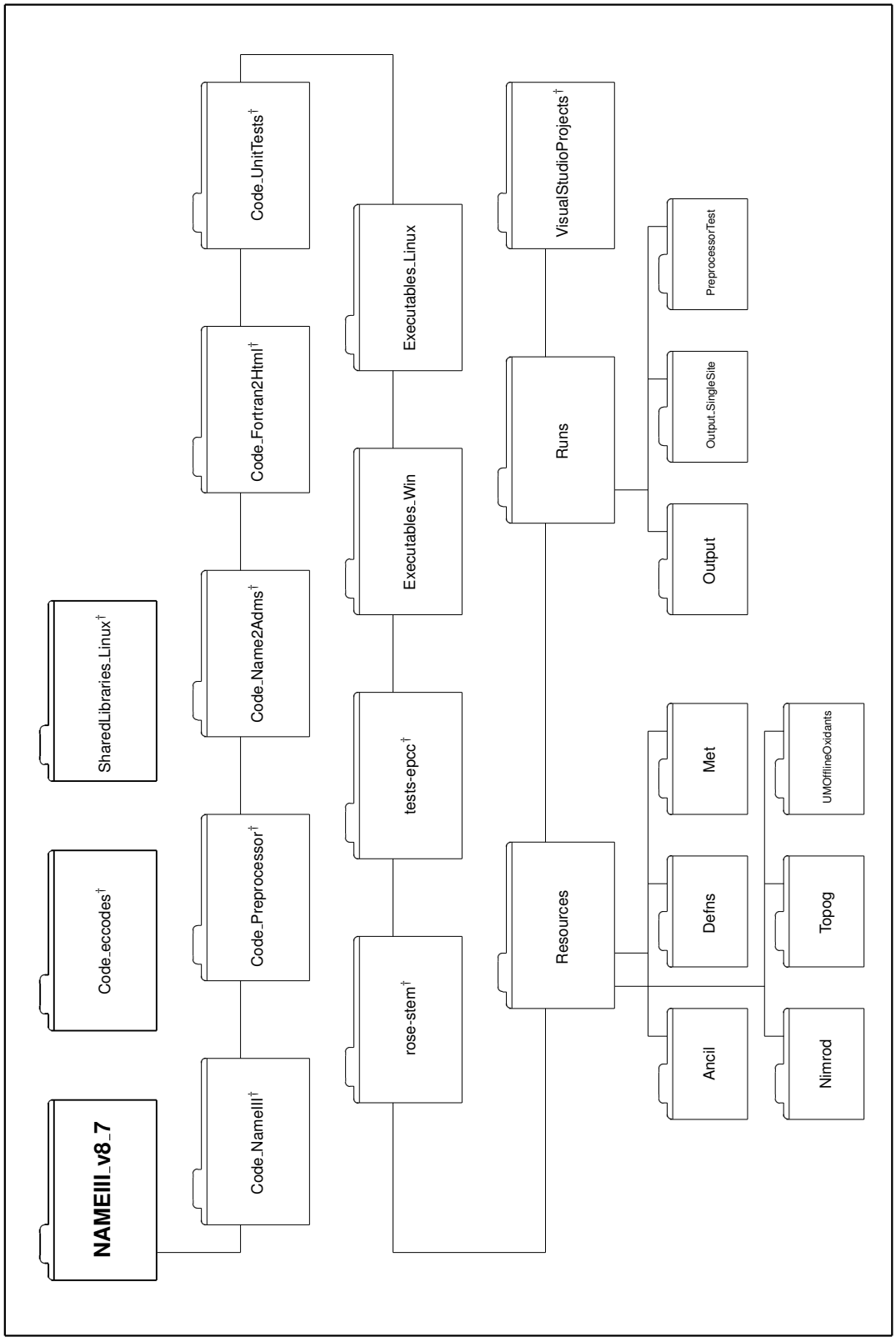


Figure 1: A schematic of the folder structure of the NAME distribution. Folders marked with a dagger† are only present when the source code has been supplied with your distribution.

via an MPI launcher command, typically called *mpirun* or *mpiexec* (though the precise command name and options will be platform specific). The launcher will handle the configuration for all the behind-the-scenes MPI communications. Most users find launching NAME from within a script to be the most useful approach. This is also the way we usually run NAME within the Met Office. See MOSRS page on *"Compiling, Running and Debugging NAME"* for further details.

The NAME training material, available to download from MOSRS, provides some example input data for testing your NAME installation. Alternatively, there are some example NAME input files in the **Runs** subfolder of the NAME distribution (though the supporting NWP met data for these examples is not stored as part of the distribution).

Note that the **Test_Runs** *folder previously supplied in earlier versions of NAME was removed at v8.0.*

6 Troubleshooting problems with NAME

Some of the information in this section is quite old, but the content is still mostly valid. However, further troubleshooting information is now available in the "NAME Troubleshooting" page on MOSRS.

A number of common problems have occasionally been encountered by external users when they try to run NAME. These are described in the troubleshooting guide on the following pages, which on encountering any problems should be consulted in the first instance. Please feedback any further items which you think would be useful to add to this list.

Problem	Possible solution
On Linux systems, NAME fails at launch with a memory fault	The most likely cause of this problem is an insufficient size for stack memory. This can be corrected by using the **ulimit** command to increase the stack size, viz. • **ulimit -s unlimited**, or • **ulimit -s** N for an explicit size N. The stack size should be specified prior to invoking NAME, e.g. within a script, or alternatively it may be declared within a user's `.profile` or `.bashrc file`. Similarly, when running with multiple threads under OpenMP, it is recommended to set the environment variable `OMP_STACKSIZE` using `export OMP_STACKSIZE='64m'` This defines the stack size that is allocated for any additional threads created during the NAME run. The default, if not set explicitly, is 4 MB ('4m') but this might not be sufficient for running NAME under OpenMP and the NAME run will fail with a 'memory fault' or 'segmentation fault' if the stack size limit is exceeded by any thread during the run.
Under Windows, NAME fails at launch with a system error message stating that `libguide40.dll` could not be found (or similar messages for other libraries that might be missing)	Missing .dll files can be an issue when running with Open MP on a Windows machine that does not have the Intel Fortran compiler installed (because some of the required Intel library files are not installed as standard). The problem can be resolved by downloading and installing the redistributable package that includes the required runtime .dll from the Intel website at `http://software.intel.com/en-us/articles/redistributable-libraries-of-the-intel-c-and-fortran-compiler-for-windows/`
NAME reports a problem with the input file even though it appears to be correct	Check for use of TAB characters in the NAME input files. TAB characters should **not** be used in any NAME input file as they can have unpredictable consequences. Hint: TAB characters can be highlighted in the editor `NEdit` by using the `Preferences > Apply Backlighting` option. Problems can also arise when copying files between Windows and Linux platforms due to a format difference in text files (Windows files contain an extra 'line-feed' at the end of each line). Some transfer mechanisms automatically convert the format (e.g. ftp) but others may not. Any problems can be resolved by explicitly converting the file format using the pair of Linux utilities **dos2unix** and **unix2dos**. The file format issue also affects single-site met files (e.g. if they were originally created from an Excel spreadsheet, say).

NAME successfully reads and checks the input file(s) but fails when attempting to read an NWP topography file	This is possibly an endian issue where the native endian on your system is different to the endian used by the NWP met and topography data (this causes an error when attempting to read such files in unformatted I/O statements). Big endian format is used for NWP meteorological data from the Met Office's Unified Model. Endian conversion is performed automatically for some compilers (e.g. more recent versions of the Intel compiler) but is not supported for all compilers. Endian conversion can be invoked explicitly by setting the environment variable `F_UFMTENDIAN`, e.g. • `F_UFMTENDIAN = "big"` to set the default behaviour of the system to apply little-endian to big-endian conversion on all logical units, or • `F_UFMTENDIAN = "big;little:N"` to apply conversion on all logical units with the exception that endian conversion is not performed on logical unit N. In practice, the user should set this environment variable with **F_UFMTENDIAN = "big;little:111"** to ensure the correct behaviour appropriate to the NAME model. The reason for reserving unit number 111 on which the conversion is not done is associated with *restartable* runs. The information contained within NAME restart files includes some derived types. However, little-endian to big-endian conversion is not supported for derived types in unformatted I/O. The restart file (reserved unit 111) must therefore be written in the native endian. a) On linux systems, type **export F_UFMTENDIAN="big;little:111"** to set the environment variable prior to running the NAME executable. b) On Windows, type **set F_UFMTENDIAN=big;little:111** at the DOS prompt or in a batch file prior to calling the NAME executable. The above command appears to be sensitive to case on different versions of the Windows operating system, and users may need to experiment with lower case and upper case variants of this text.

NAME is unable to read gzipped NWP met data files	The current version of NAME is not able to read gzipped met files directly (although the intention is to explore the possibility of including this as an option in the future). Instead, all NWP met data files need to be unzipped before they are read by NAME. When using a met restore script with NAME, the met restore process needs to ensure that each met file is restored in an unzipped form (typically, this might involve copying a gzipped file from a met archive to a local met folder and then unzipping that local duplicate). When reading NWP met data directly from a met folder (i.e. not using a met restore script), the user should ensure that all the required met files are available unzipped prior to running NAME. a) On linux systems, file zipping/unzipping is done using the general Linux file utility **gunzip**. You should ensure that the gunzip utility is available on your system (although this should be part of any standard Linux build). b) On Windows, file zipping/unzipping can be done using the WinZip utility. Note that NWP met files from the Met Office's Unified Model used standard compression (-5 option in gzip) prior to 2009, and high compression (-9 option) between 2009 and 2017. Since 2017, UM met files are no longer gzipped but instead use an in-house packing algorithm that NAME is able to understand via use of the Shumlib external libraries.
A met restore script cannot unzip the gzipped data files	Ensure that a suitable unzip utility is available on your system. a) On linux systems, use the general file utility **gunzip** (which should be part of any standard Linux build). b) On Windows, use the WinZip utility.

For the ecCodes version of NAME (available for Linux only):
 	
NAME is unable to find the ecCodes shared object libraries at run time and reports the error message *"error while loading shared libraries: libeccodes_f90.so: cannot open shared object file: No such file or directory"*.
	The location of the ecCodes .so libraries needs to be added to the environment variable `LD_LIBRARY_PATH` before running the ecCodes version of NAME, e.g.,
	`export LD_LIBRARY_PATH=`
	`${LD_LIBRARY_PATH}:${ECCODES_DIR}/lib`
	where `${ECCODES_DIR}` is the top-level directory of the ecCodes installation.
	Similarly, when reading or writing NetCDF files, the NetCDF API libraries need to be included in the `LD_LIBRARY_PATH` search path.

For the ecCodes version of NAME (available for Linux only):
 	
ecCodes does not correctly interpret the contents of GRIB format met files.
	The following environment variables need to be set to correctly configure ecCodes at run time:
	`export ECCODES_DEFINITION_PATH=`
	`${ECCODES_DIR}/share/eccodes/definitions`
	`export ECCODES_SAMPLES_PATH=`
	`${ECCODES_DIR}/share/eccodes/samples`
	where `${ECCODES_DIR}` is the top-level directory of the ecCodes installation.
	The environment variable `ECCODES_DEFINITION_PATH` can be set to use local definition files in preference to the standard definition files provided with the ecCodes distribution. To do this, set the environment variable as
	`export ECCODES_DEFINITION_PATH=`
	`${ECCODES_DIR}/local/definitions:${ECCODES_DIR}/share/eccodes/definitions`
	when the local definition files are stored under the `/local` subdirectory of the ecCodes installation (or modify the first path if the local files are stored under a different location).