Ash3d: Documentation and User's Guide

Hans F. Schwaiger, Larry G. Mastin, Mike Randall, and Roger Denlinger

June 30, 2021

Contents

Abstract

Atmospheric dispersal of volcanic ash represents the most widespread of volcanic hazards and the hazard with the greatest potential for economic impact. The economic effects of ash distribution were demonstrated during the 2010 eruption of Eyjafjallajökull volcano in Iceland, when flight cancellations and delays throughout Europe caused billions of dollars in economic loss to airlines and travelers [8]. This event also demonstrated that volcanic hazards are not limited to areas near the volcano. Flight cancellations and delays during the 2011 eruption of Puyahue/Cordon Caulle in Chile extended halfway around the world to Australia and New Zealand, for example.

Chapter 1

Introduction

In 2012, we introduced Ash3d, a volcanic ash transport and dispersion model used to forecast ash-cloud movement and ash deposition during eruptions [19]. This was followed in 2013 by the introduction of a web interface that allowed any user with an account to simulate ash-cloud transport and deposition [13]. Subsequently, we have used Ash3d to study ash transport during recent eruptions [17], to evaluate the sensitivity of model inputs to outcomes [**?**], examine effects of umbrella-cloud growth on eruption dynamics [14, 15], study whether and how ash from New Zealand supereruptions could be deposited in Antarctica [6], develop probabilistic tephra-hazard analyses for eruptions from Mount St. Helens [16] and Taupo, New Zealand [1], and examine Bayesian ensemble techniques to improve ash-cloud forecast accuracy [5].

Development of a web interface allowed users all over the world to use the model without downloading and compiling the software or downloading large amounts of numerical weather prediction (NWP) model data used to run the model. The web interface however provides only a small subset of functions available in Ash3d. Specifying a grain-size distribution, or a time series of erupted pulses and plume heights and eruption rates, or allowing for the growth of umbrella clouds, has not been available due to our desire to keep things simple. The software has also not been available for download, meaning that only model developers and their close collaborators have had access the full suite of features.

With this release of Ash3d source code, users can take advantage of Ash3d's full functionality. Users with particular skill or interest may also advance the source code to consider physical processes or numerical advances not yet envisioned. The complexity of this model requires that it be installed by users who are versed in Linux and comfortable running models at the command line. Users who wish to use current or forecast meteorological data may also have to have a Linux system that can run scheduled NWP data downloads (tens of Gb daily) and other updates. If you have these skills and resources, and a desire to simulate tephra transport in a 3D wind field, this code may be for you. Users need have no special knowledge in numerical methods or modeling.

We begin with a brief overview of Ash3d, describe simulations using an ASCII input file, and provide some guidance on post-processing the results. Use of Ash3d through the USGS web-based interface is briefly summarized, but is more formally presented in a companion

document [17]. Ash3d is still in development, hence its capabilities and documentation are likely to be revised in coming years.

1.1 Model Overview

Ash3d calculates the transport of volcanic ash by dividing the atmosphere into a three-dimensional grid of cells (Figure 1.1) and calculating the flow of mass through cell walls. During the eruption, tephra is injected at a constant rate into the column of cells above the volcano (Figure 1.1). Using a 3-D time-dependent wind field imported from a numerical weather prediction model, downwind advection and diffusion of ash is numerically calculated with a diffusion rate determined by a user-specified diffusivity. Individual ash particles fall at a rate determined by their settling velocity in air, and deposit when they reach the ground surface.

Figure 1.1: Example model grids

1.1.1 Initial conditions

Ash3d does not calculate the dynamics of a rising plume. Instead, it injects tephra into a column of nodes above the volcano (Figure 1.1). Users may specify that the ash be concentrated in a single cell, distributed evenly throughout the column, distributed along a user-specified vertical profile, or distributed vertically following the Suzuki equation (Figure 1.1) [3, 21].

$$\frac{\mathrm{d}S}{\mathrm{d}z} = S\frac{k^2\left(1 - z/H\right)\exp\left[k\left(z/H - 1\right)\right]}{H\left[1 - (1+k)\exp\left(-k\right)\right]} \tag{1.1}$$

where S is the total mass of erupted material in a given time step at a given particle size, H is the total plume height, z is a given elevation in the plume, and the shape factor k is an adjustable constant that controls ash distribution with height. A low value of k gives a roughly uniform distribution of mass with elevation, while higher values concentrate mass near the plume top (See Figure 1.2).

For eruptions with a larger mass-eruption rate, limiting these source terms to just the column above the vent can be inadequate and the radial spreading of an umbrella cloud must be accounted for. The radius of the umbrella cloud at a particular time is given by:

$$R = \left(\frac{3\lambda N Q}{2\pi}\right)^{\frac{1}{3}} t^{\frac{2}{3}} \tag{1.2}$$

where the volume influx rate, Q, is given by:

$$Q = C\sqrt{k_e}\frac{\dot{M}^{\frac{3}{4}}}{N^{\frac{5}{4}}} \tag{1.3}$$

The radial spreading velocity within the umbrella cloud ($r < R$) can be expressed as

$$u(r) = \dot{R}\left(\frac{3R}{4r} + \frac{r}{4R}\right) \tag{1.4}$$

Figure 1.2: Source examples

During an eruption, Ash3d injects tephra into these nodes at a constant rate. The grain-size distribution inserted into each node in the column is the same at each elevation and for each eruptive pulse. Default values are described in Chapter **??**.

1.1.2 Transport

Ash3d solves for the conservation of mass in each cell by tracking the mass concentration q with time t,

$$\frac{\partial q}{\partial t} + \nabla \cdot [(\mathbf{u} + \mathbf{v_s})\, q] - \nabla \cdot (\mathbf{K} \nabla q) = S \tag{1.5}$$

where $\mathbf{u}$ is the 3-D wind vector, $\mathbf{v_s}$ the settling velocity, K is the diffusivity, and S is a source term. For most source types, S is non-zero only in the column of nodes above the volcano. The exception is the umbrella cloud source term which includes a 3×3 group of cells centered over the vent. The settling velocity is determined via [2, p.182]

$$v_s = \sqrt{\frac{4 d \rho_p g}{3 C_d \rho_a}} \tag{1.6}$$

where d is the particle diameter, ρ_p is the particle density, g is the acceleration of gravity, C_d is the drag coefficient, and ρ_a is the density of air. Ash3d can use different models of C_d to account for non-spherical particles. By default, Ash3d calculates the settling velocity of each particle size using equations of Wilson and Huang [23] for non-spherical particles.

$$C_d = \frac{24}{\mathrm{Re}} F^{-0.828} + 2\sqrt{1.07 - F} \tag{1.7}$$

Re is the Reynolds number and F is a shape factor (see Appendix D for more details.). Alternative models that can be used include an optional Cunningham slip correction factor [20, p.407], which can increase the fall rate for small ($< 30\,\mu$m) particles at high elevation; Stokes flow; and modifications to Wilson and Huang by [7] and [18].

The advection term of Eq. 1.5 (the second term on the left-hand side (LHS)) is calculated explicitly using the Donor Cell Upwind method of solution (see [19] for details). The diffusion term (third LHS term) is calculated implicitly at the end of each time step. The diffusivity, K, can be spatially variable and can be a function of the local meteorological conditions. We have found that modeled clouds match well with observed ones when $K = 0$, and have set it to this value for simulations using the web interface.

1.1.3 Deposition

Ash3d tracks the mass flux of each grain size across cell boundaries and accumulates a deposit once tephra falls through the cell boundary that represents the ground surface at a particular location.

Chapter 2

Installation

Ash3d was developed to run on a Linux system and has been tested on several popular Debian and Red Hat-based distributions, running on hardware as light as Intel Atom processors or 32-bit ARM systems (Raspberry Pi 3), common personal computer hardware, as well as modern high-performance computing environments. It has been successfully built on other Unix varieties such as OpenBSD and Darwin (MacOS). Experimental limited installations on Window 10/11 have been successful, though implementing the full functionality is on-going.

Library dependencies Ash3d requires three auxiliary libraries:

- libprojection.a (https://code.usgs.gov/vsc/ash3d/volcano-ash3d-projection)
  This library converts between longitude/latitude coordinates and several projections commonly used in NWP files. This repository is mirrored at https://github.com/DOI-USGS/volcano-ash3d-projection.
- libHoursSince.a (https://code.usgs.gov/vsc/ash3d/volcano-ash3d-hourssince)
  This library converts dates and time values to the number of hours since a reference year. This repository is mirrored at https://github.com/DOI-USGS/volcano-ash3d-hourssince.
- libmetreader.a (https://code.usgs.gov/vsc/ash3d/volcano-ash3d-metreader)
  This library is an interface between a calling program and a wide variety of numerical weather prediction (NWP) models as well as radiosonde data. This repository is mirrored at https://github.com/DOI-USGS/volcano-ash3d-metreader.

Each of these libraries were developed as a part of Ash3d, but were split into separate libraries so as to support other software projects requiring an interface to NWP models. Instructions for building and installing these libraries are given in the respective repositories. The default installation location for these libraries is `/opt/USGS`. This could be changed to suit your system by editing the `makefile` (`INSTALLDIR=~/USGS` for example), but should be consistent since the Ash3d `makefile` expects a consistent location. A minimal installation of these libraries only requires a Fortran compiler, however only ASCII files of wind data (radiosonde or other profiles) would be available. To read NWP files from NCEP, NASA,

ECMWF, WRF or otherwise, MetReader must be compiled with either NetCDF (preferably v4) or the GRIB library ecCodes.

This can be installed for a Red Hat-based system (RHEL, CentOS, Rocky, Fedora) by

```
sudo yum install netcdf netcdf-devel eccodes eccodes-devel
```

or for Debian (Ubuntu, Mint, Rasperian)

```
sudo apt-get install netcdf netcdf-devel eccodes eccodes-devel
```

In the MetReader library, it is encouraged to also download and install `netcdf-java` which is used to convert GRIB files to NetCDF https://www.unidata.ucar.edu/software/netcdf-java To use this utility, you will need a java installation:

```
sudo yum install java-openjdk
```

or for Debian (Ubuntu, Mint, Rasperian)

```
sudo apt-get install default-jdk
```

Although not needed for a minimal Ash3d installation, the `lapack` and `blas` libraries are required if the Crank-Nicolson scheme is selected for calculating diffusion.

So to build the needed libraries, run:

```
cd /to/build/directory/
git clone https://github.com/DOI-USGS/volcano-ash3d-hourssince
cd volcano-ash3d-hourssince
  For local changes; vi makefile
make all
make check
[sudo] make install

cd /to/build/directory/
git clone https://github.com/DOI-USGS/volcano-ash3d-projection
cd volcano-ash3d-projection
  For local changes; vi makefile
make all
make check
[sudo] make install

cd /to/build/directory/
git clone https://github.com/DOI-USGS/volcano-ash3d-metreader
cd volcano-ash3d-metreader
  For local changes; vi makefile
make all
make check
[sudo] make install
```

Building Ash3d Once the dependent libraries have been installed, Ash3d can be downloaded from https://code.usgs.gov/vsc/ash3d/volcano-ash3d.

An autoconf installation process is in development, but currently, compilation is controlled through a manual editing of `src/makefile`.

The top block of makefile contains all the variables that the user should need to set:

- `SYSTEM = gfortran`
  If you want to add blocks for different compiler flags, you can control the paths through this variable. Tested compilers include `gfortran`, `ifort`, and `aocc`. Compiler paths, flags, options are all specified in include files with names `make_[SYSTEM].inc`

- `RUN = OPT`
  This variable allows easy switching among different compiler flags for debugging (`DEBUG`), profiling (`PROF`), and optimized (`OPT`). Additionally, `OPTOMP` can be used to include openMP directives.

- `OS=LINUX`
  This variable specifies the operating system class used. Default is `LINUX`,but can also be `MACOS` or `WINDOWS`. This is mainly used to direct the software on how to build local paths and execute commands.

- `USGSROOT=/opt/USGS`
  This is the path to where libhoursince, libprojection, and libmetreader are installed.

- `ASH3dCCSRC=./`
  This is the location of the `src` directory. The build directory can be elsewhere.

- `INSTALLDIR=/opt/USGS/Ash3d`
  This is the install path.

- `USENETCDF = T`
  This variable is used to toggle (`T` or `F`) inclusion of NetCDF functionality. MetReader would also need to be compiled with a consistent flag.

- `USEGRIB = T`
  This variable is used to toggle (`T` or `F`) inclusion of GRIB functionality. MetReader would also need to be compiled with a consistent flag.

- `USEPOINTERS = F`
  This variable allows some variables to be declared as allocatable pointers instead of allocatable arrays. Declaring variables as allocatable pointers allows an easier interface with C programs (such as ForestClaw). The option to keep variables as allocatable arrays is primarily for compatibility with some older compilers. The current implementation of OpenMP seems to require that pointers not be used. This variable will likely be deprecated in favor of exclusive allocatable pointer variables.

- `USEEXTDATA = F`
  This variable allows (if `T`) some external data files to be read at run-time. The default behavior is that a global airport list and a global list of volcanoes with default eruption source parameters are read as data variables at compile-time to minimize the external files the executable needs at run-time. Compiling some subroutines with several thousand lines in which these data are stored can add significantly to the compilation time and memory requirements, sometimes exceeding the available resources (such as on the Raspberry Pi 3). If this variable is set to `T`, the data files in `Ash3d/share` will be copied to the installation directory. This path can be supplanted with another location using the optional environment variable `ASH3DHOME`.

- `FASTFPPFLAG =`
  This variable allow activating some limits on Ash3d computations which can significantly speed-up calculations. If `-DFAST_DT` is used, the determination of the maximum allowed time-step is limited to the time-steps of the NWP files, which are generally at 1,3 or 6 hours. Without this flag set, the maximum `dt` is calculated each time-step. This might be necessary for some processes such as the radial spreading of umbrella clouds. A second flag that can be used is `-DFAST_SUBGRID` which limits the flux calculations to just the bounding box of the current airborne ash cloud.

- `USEPLPLOT = F`
  `USEDISLIN = F`
  `USEGMT = F`
  These allow the inclusion of various plotting libraries to be build into the Ash3d post-processing tools, allowing the direct creation of plots from `Ash3d_PostProc`. PLplot is often available as a distribution package and can be installed via: `yum install plplot plplot-devel plplot-fortran-devel`. Some backward-incompatibilities were introduced in version 5.13 so please use this version or newer. DISLIN is another graphics package that allow direct creation of plots, but has the added advantage of access to contour lines needed for creating shapefiles. Generic Mapping Tools (GMT) is typically used through temporary control files and system calls. Set `USEGMT` to `T` only to use the GMT Fortran bindings to the API. This currently is not fully functional.

- `LIMITER = LIM_SUPERBEE`
  This variable controls which limiter is used in the advection routines. The superbee limiter performs very well and is the default, however other limiters can be used if desired (e.g. if linearity is needed). Available options are: no limiter (`LIM_NONE`), Lax-Wendroff (`LIM_LAXWEN`), Beam-Warming (`LIM_BW`), Fromm (`LIN_FROMM`), minmod (`LIM_MINMOD`), superbee (`LIM_SUPERBEE`), and MC (`LIM_MC`).

- `DIFFMETH=CRANKNIC`
  This variable controls the integration scheme used for diffusion. The default is `CRANKNIC` which invokes the Crank-Nicolson semi-implicit scheme, but this requires the system libraries `libblas` and `liblapack` to be installed. Alternatively, if set to `EXPLDIFF`, the explicit scheme is used.

- `PII=ON`
  Ash3d does an initial system check to determine the current working directory, location of any shared files, tests of compiler (and version) and general computing environment, including the user name and host name. If this flag is set to `ON`, this information is collected and included in the output file. This information can be useful as archival metadata on run output, but it may also be considered insecure to have usernames, hostnames and complete system paths. Set to `OFF` to disable this logging.

Once these variables are modified to your system and preferences, Ash3d can be built by typing `make`. To install, type `make install` either as root or with a installation path for which you have write privileges.

To uninstall, type `make uninstall`.

Chapter 3

Usage

Ash3d

3.1 Preliminary meta-data

Wind files
Volcano data
Airport/POI data

3.2 Running Ash3d on the command-line

The Linux command-line provides a more versatile but less user-friendly environment for running Ash3d simulations. We do not offer support for installing or running Ash3d on computers outside the USGS, but we are willing to work with collaborators who wish to run Ash3d in a more advanced environment than is possible through the web interface. Below, we explain how to set up and run a simulation in a Linux environment. Ash3d is in a state of continuing development, and we recommend that potential users consult the authors for updates before following these instructions.

Figure 3.1: Examples of model output

3.2.1 Model input using an ASCII input file

The Ash3d executable reads from an ASCII input file that supplies information on source parameters, output file types, and other options. A file for a simulation at Redoubt volcano is illustrated in Appendix A. If these files are viewed in a Linux text editor such as `vi` or `gedit` that converts syntax elements for a shell script into colors, the same color scheme will be visible, allowing users to easily discriminate comments from parameters. The input file is further divided into blocks, delimited with lines of asterisks. Each of these blocks is described below.

Block 1 : Model grid
The lines in this block primarily define the size, shape, and location of the model domain.

```
**************************************************************************
Spurr                                # Volcano name
1 4 -107.0 50.0 50.0 50.0 6367.47    # Proj flags and params
-154.0 58.0                          # x, y of LL corner of grid (km, or deg.)
20.0 6.0                             # grid width and height (km, or deg.)
-152.251 61.299 2.309                # vent location         (km, or deg.)
0.25       0.25                      # DX, DY of grid cells  (km, or deg.)
0.5                                  # DZ of grid cells      (always km)
0.0     4.0                          # diffusion coefficient (m2/s), Suzuki constant
1                                    # neruptions, number of eruptions or pulses
**************************************************************************
```

Line 1 gives either the volcano name or the volcanos Smithsonian ID. If the number begins with 0 or 1, then the volcano database used is the Catalog of Active Volcanoes of the World [Siebert et al., 2010]. If the number does not begin with 0 or 1, then the Volcanoes of the World (VOTW) database is used [Global Volcanism Program, 2013. Volcanoes of the World, v. 4.9.4 (17 Mar 2021). Venzke, E (ed.). Smithsonian Institution. Downloaded 13 Apr 2021. https://doi.org/10.5479/si.GVP.VOTW4-2013.]. If the CAVW number is given, Ash3d looks up the eruption source parameters (ESP) for this volcano in the ESP spreadsheet of [12]. In the example file, the volcano name is given.

Line 2 gives the projection parameters for this simulation. Ash3d uses the `volcano-ash3d-projection` package to specify the projection and to calculate any needed coordinate transformations. Documentation for this package can be found at https://github.com/DOI-USGS/volcano-ash3d-projecti These parameters on Line 2 define the projection for the computational grid and can be different than the projection used for the numerical weather prediction (NWP) files. Some of those NWP models, such as the NOAA National Center for Environmental Predictions (NCEP) Global Forecast System (GFS) model, are run on a spherical earth, hence the wind vectors are given in a 3-D grid of longitude, latitude, at pressure levels. Other models, such as NOAAs North American Model are run using a grid that is projected onto a planar coordinate system. After Ash3d reads the NWP atmospheric data, it must know the type of projection in order to find the portion of the NWP model grid that lies within the Ash3d model domain. Ash3d uses the MetReader library to read the atmospheric data and to interpolate values from the NWP grid onto the computational grid used by Ash3d. MetReader also projects the wind vectors onto the coordinate system specified by this line. The types of NWP model output that Ash3d can read, and their projection parameters, are given in Table 3.1. The first two parameters on this line are `latlonflag` and `projflag`, where:

- `latlonflag` is an integer whose value is 0 if coordinate system is projected, 1 if coordinates are in longitude and latitude
- `projflag` is an integer that indicates the projection type

Numbers that follow these two integer flags are the projection parameters that are different for different map projections. Below are some examples of this input line. The comment in

latlonflag	projflag	Description	Expected parameters
0	0	Non-geographic Cartesian grid	N/A
0	1	Polar Stereographic	λ_0 ϕ_0 k_0 R_e
0	2	Albers Equal Area	λ_0 ϕ_0 ϕ_1 ϕ_2 R_e
0	3	UTM	Not yet functional
0	4	Lambert Conformal Conic	λ_0 ϕ_0 ϕ_1 ϕ_2 R_e
0	5	Mercator	λ_0 ϕ_0 R_e
1	N/A	Longitude/Latitude	N/A

Table 3.1: Ash3d projection options

blue following the hash mark (#) gives the projection type.

```
0 1 -135.0 90.0 0.933 6371.229 #Polar stereographic
```

Parameters 3-6 are λ_0, the longitude of the projection point; ϕ_0, the latitude of the projection point; k_0, the scale factor at the projection point; and R_e, the Earth radius used in kilometers.

```
0 4 -95.  25.0 25.0 25.0 6371.229 #Lambert Conformal Conic
```

Parameters 3-7 are λ_0, the longitude of the projection origin; ϕ_0, the latitude of the projection origin; ϕ_1, the latitude of the first secant; ϕ_2, the latitude of the second secant; and R_e, the Earth radius used.

Line 3 gives the x and y (or longitude and latitude if `latlonflag = 1`) values of the lower left corner of the grid. These values must be in the same coordinate system defined by the projection parameters on line 2.

Line 4 gives the model domain width and height, in degrees if longitude and latitude are used, or kilometers if a projected coordinate system is used. If longitude and latitude are used and if the width is given as 360°, then Ash3d uses a periodic grid.

Line 5 gives the vent location, also in the same units as the specified coordinate system. This line may take either two or three values. The first two values are required and correspond to the x and y (or longitude and latitude) coordinates of the vent. A third, optional value gives to the vents elevation in kilometers. If no elevation is given, Ash3d assigns the volcano an elevation equal to that of the topography at this location, or zero if topography is not used in this model run. For the web interface, topography is not used.

Line 6 gives the horizontal grid spacing in the model, in kilometers if a projected grid is used, or degrees if longitude/latitude are used. If the width and height of the model domain are not an integral number of cell distances, the location of the upper right corner of the model domain is adjusted to be an integral number of cell distances from the lower left corner.

Line 7 specifies the vertical grid structure to be used. If a number is given, it is interpreted to be the constant Δz for a regular vertical grid spacing. The units of this input parameter are always kilometers. Alternatively, a text string can be given which specifies different options for a variable spacing of the vertical grid. This text string must be one of the following:

- `dz_plin`: for piecewise linear
- `dz_clog`: for constant Δz in $\log z$

- `dz_cust`: for custom

For each of these variable-Δz cases, an additional line must be provided in the input file immediately after Line 7. For `dz_plin`, this line must start with the number of linear segments, followed by the number of cells and Δz in each segments (n_{seg} n_{z_1} Δz_1 n_{z_2} $\Delta z_2 \cdots$ n_{z_n} Δz_n). For `dz_clog`, the additional line just contains z_{top} followed by the number of cells in z. For `dz_cust`, the additional line must start with the vertical number of cells nz, followed by a list of nz values for Δz.

Line 8 parameters specify the diffusion coefficient (K) and the vertical distribution of mass in the column, respectively. The diffusion coefficient is specified in m^2/s and is applied to both the horizontal and vertical diffusivities. If K=0.0, Ash3d disables the diffusion calculation. Note that unless Ash3d has been compiled with the Crank-Nicolson scheme invoked (which requires the `lapack` library), including diffusion can significantly increase the computation time through the time-step constraints with explicit diffusion solvers.

The vertical distribution of mass may be specified either as a number, or as a text string: `point`, `line`, `profile`, `umbrella`, and `umbrella_air`. These possibilities are illustrated in Figure 3.2. If the input is:

- a number: it is assumed to be the Suzuki constant k in Equation 1.1. Mass distributions for different k are illustrated in Figure 3.2a.
- `point`: all mass is placed in a single cell at the plume top (Figure 3.2a).
- `line`: mass is distributed evenly from the vent elevation (if provided) or from sea level (if vent elevation is not provided) to the plume top (Figure 3.2a)
- `profile`: each eruptive pulse line in Block 2 must be supplemented by an additional line specifying the profile, as described below.
- `umbrella` or `umbrella_air`: the plume height in Block 2 is interpreted as the height of the umbrella cloud (ash in the overshooting top, above the umbrella cloud, is assumed to collapse gravitationally back into the umbrella rather than being advected horizontally by winds at that altitude). This source type is most appropriate for eruptions of VEI=6 and larger, and may be appropriate for some VEI 4 or 5 eruptions [**?**]. Source nodes for these cases consist of a column of nodes extending from the vent elevation (if provided) to 75% of the umbrella top height, or from sea level (if vent elevation is not provided) to 75% of the umbrella top height. From 75% to 100% of the umbrella top height, source nodes consist of a matrix 3×3 in plan view (Figure 3.2b). Within the height range of the 3×3 matrix, from the vent outward to the edge of the expanding umbrella cloud, radial wind vectors are added to the ambient wind field. The magnitude of the radial wind vectors is calculated using corrected Equations (2) and (3) of [**?**], and is proportional to the mass eruption rate. The only difference between `umbrella` and `umbrella_air` is that the latter is used only for simulations of airborne ash, where a single grain size is used, and the volume of ash is assume to equal 5% of the total erupted volume. Only one eruptive pulse may be specified when the umbrella source is used.

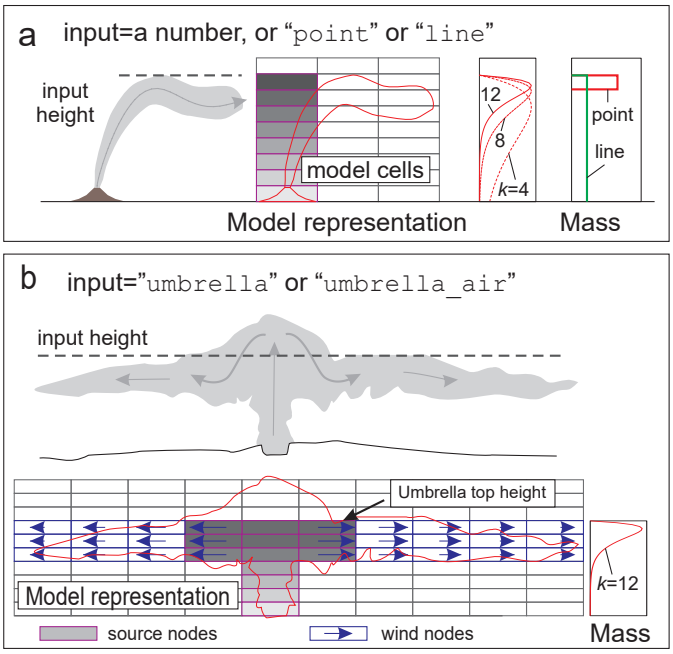


Figure 3.2: Illustration of vertical mass distributions that can be specified in Ash3d, and the association of these inputs with other aspects of model setup, such as plume height and the configuration of source nodes.

- neither a number nor one of the above types: Ash3d assumes this source type is a user-specified custom source type. These are described in detail in section **??**

Line 9 gives the number of eruptions or eruptive pulses in the simulation. The eruptions or eruptive pulses may be either contiguous or non-contiguous in time. Their duration should not be shorter than the average time step in the model (about a minute).

Block 2 : Eruptive pulses
For the standard source types (`suzuki`, `point`, `line`, `umbrella`, or `umbrella_air`), the number of lines in this block equals the number of eruptions or eruptive pulses specified in Line 9 of Block 1. A line of input for these types contains three integers followed by four real numbers, for example:

```
**********************************************************************************
1992 08 19   1.0   3.5     13.7   0.014
**********************************************************************************
```

The first four numbers represent the year, month, day and hour (UTC) of the start of the eruption. The following three numbers represent the eruption duration in hours, the plume height in kilometers above sea level, and the erupted volume in cubic kilometers dense-rock equivalent (DRE). Ash3d converts the erupted volume to a mass assuming a magma density of 2500 kg/m^3. This can be changed at run-time is desired (See Section **??**). If the year is zero as in the example input file, the model is run in forecast mode where the hour is interpreted as the number of hours after the start time of the wind files. In addition, if the duration, plume height, or erupted volume is negative, it is replaced with the default ESP value for that volcano from the spreadsheet of [11].

If the source type is `profile`, then the eruption lines are read as three integers, five real numbers, and an integer where two numbers are the Δz and number of z-points defining the profile. This type of source type must have a line following with the list of fractional values for each of the Δz segments (should total unity). An example of this type is given below.

```
**********************************************************************************
1992 08 19   1.0   3.5  0.1  13.7   0.5 14
0.0 0.1 0.1 0.1 0.1 0.2 0.2 0.0 0.0 0.0 0.05 0.05 0.1 0.0
**********************************************************************************
```

Figure 3.3: Example of source term options

If the source type is not recognized, Ash3d just reads the first line of this block which must have at a minimum three integers for the year, month and day, followed by three real values for the hour, duration and height. There may be more values on the eruption source line and potentially additional lines to read for the custom sources, but this is parsed later after the remaining blocks are read.

Block 3 : Wind files and simulation time
Line 1 of Block 3 gives at least two integers, `iwind` and `iwindformat`. The first number (`iwind`) specifies the structure of the windfiles and can have the following values: 1=1-D wind sounding(s), 2=3-D gridded ASCII files, 3=a single NWP file (all variable data for all steps in one file), 4=multiple NWP files (all variable data for one step in individual files), 5=non-standard files that require hard-wired paths.

The distinction between `iwind`=3 and `iwind`=4 has been relaxed and these are now interchangeable. Ash3d attempts to read two more integer values on Line 1 of this block: a grid ID, and a format code. The grid code is the NCEP grid ID described at http://www.nco.ncep.noaa.gov/pmb/docs/on388/tableb.html and the format code must be either 1 (ASCII), 2 (NetCDF), or 3 (GRIB v1 or v2). If these values are not provided, the grid code is assigned if known from the NWP product given by `iwindformat` and the data format code is assumed to be 2=NetCDF. The combinations of `iwind` and `iwindformat` are shown in Table 3.2.

If `iwind`=1, Ash3d reads from one or more ASCII files of a 1-D wind sounding. This option can be used if no numerical weather prediction data are available or if data from radiosonde measurements is deemed more reliable than NWP model results. If `iwindformat`=1, then Ash3d expects the user-provided files to have a format of 3 header lines, followed by three or more columns of data. The format specification is documents in the MetReader User's Guide, but also outlined in the Appendix B.1.

If `iwindformat`=2, then Ash3d expects the 1-D profile to be from the global radiosonde data available from https://ruc.noaa.gov/raobs/ or from http://weather.uwyo.edu. Details on the particular formats Ash3d is able to read is described in Appendix B.1.

Since there is no grid for these data, `igrid`, if provided, is interpreted to be the number of stations with 1-D data. The number of windfiles must be an even multiple of `igrid`.

When `iwind`=2, Ash3d reads a 3-D atmospheric fields in ASCII format that is converted from NetCDF format using a Java script that was originally written before Ash3d was modified to read NetCDF files directly. This option is no longer used. `iwind`= 3 or 4 indicates that the atmospheric files contain 3-D time-series data. Originally, `iwind`=3 was used for indicating that all data were stored in a single file with multiple time steps, and `iwind`=4 was for multiple NetCDF files, each for one time step. This distinction is deprecated as currently, all data files provided with `iwind`= 3 or 4 are evaluated for the time steps available. This then can accommodate the occasional NWP products that provide data in a multiple-file, multiple-time-step format.

For some reanalysis products that do not store data in the above formats, such as the products that store a month or a year of data for only one variable in a single file (e.g. NCEP NCAR 50-year reanalysis, `iwind`=5 can be used. For these cases, Ash3d expects the NWP files to have specific filenames and to be stored in a specific directory structure. Details for these products are described in the MetReader documentation and summarized in Chapter B.1.

Line 2 gives the integer `iHeightHandler`, which indicates how Ash3d should respond if it finds that the plume height exceeds the highest level in the NWP model. Numerical weather prediction models give 3-D atmospheric fields with the vertical coordinate in pressure levels. For the GFS 0.5 degree model, the lowest pressure level is $\sim 40\,\text{kPa}$, which corresponds to

iwind	iwindformat	grid	data	Description
1				1-D wind sounding(s)
	1	n	1	User-specified
	2	n	1	global radiosonde data
2				3-D gridded ASCII files
				Not implemented/deprecated
3				one NWP file for all data
4				one NWP file per time step
	0		2	User-defined via template
	3	221	2/3	North American Regional Reanalysis NARR
	4	221	2/3	NAM Regional North America
	5	216	2/3	NAM Regional Alaska
	6	104	2/3	NAM N. Hemisphere
	7	212	2/3	NAM Regional CONUS
	8	218	2/3	NAM Regional CONUS
	9	227	2/3	NAM Regional CONUS
	10	242	2/3	NAM Regional Alaska
	11	196	2/3	NAM Regional Hawaii
	12	198	2/3	NAM Regional Alaska
	13	91	2/3	NAM Regional Alaska
	14		2/3	NAM Regional CONUS
	20	4	2/3	GFS
	21	3	2/3	GFS
	22	193	2/3	GFS
	23	2	2/3	NCEP-DOE Reanalysis 2
	24		2/3	NASA-MERRA-2 Reanalysis
	25	2	2/3	NCEP/NCAR Reanalysis 1
	28	170	2/3	ECMWF ERA-Interim Reanalysis
	32		2/3	Air Force Weather Agency subcenter = 0
	33		2/3	CCSM3.0 Community Atmosphere Model
	40		2/3	NASA GEOS-5 Cp
	41		2/3	NASA GEOS-5 Np
	50		2/3	WRF - output
5				files that require hard-wired paths
	25	2	2/3	NCEP/NCAR Reanalysis 1
	26	45	2/3	JRA-55
	27	2	2/3	NOAA-CIRES 20th Century Reanalysis
	29		2/3	ECMWF ERA5 Reanalysis
	30		2/3	ECMWF ERA-20C Reanalysis

Table 3.2: Ash3d options for wind data

about 50km altitude. The atmospheric conditions at higher altitude is unknown from these models. If the input plume height is higher than the highest altitude in the NWP model, Ash3d responds in one of the following ways: if `iHeightHandler`=1, Ash3d execution stops and writes an error message to standard output and the log file. If `iHeightHandler`=2, Ash3d continues execution and uses the wind vectors in the top-most pressure level in all lower pressure levels. Temperature is calculated for these higher altitudes using the US Standard Atmosphere. Ash3d also writes a warning message to standard output and to the log file.

Line 3 gives the simulation time in hours. The simulation time is assumed to start at the time of the first eruption or eruptive pulse. Ash3d checks the time duration of the atmospheric files to ensure that the entire simulation time is contained within the scope of provided files.

Line 4 is a "yes" or "no" parameter that specifies whether to stop computation when 99% of the erupted mass has left the domain either through deposition or advecting out the boundaries. Setting this parameter to "yes" allows for faster execution if users are primarily interested in simulating deposition. If simulating ash-cloud transport is of primary interest, users may prefer to set this parameter to "no".

Line 5 specifies `nwindfiles`, the number of atmospheric files to be read. Ash3d examines the time stamp of all the steps of all the atmospheric files to ensure that the start time and end time of the simulation are within the scope of the provided files.

Block 4 : Output options

Most lines in this block specify types of output options that can be written at specific output steps. Lines 1-15 require "yes" or "no" to indicate whether a particular type of output should be written. These types of output are listed in Table 3.3 along with the names of files written out. The ESRI® ASCII files noted in Table 3.3 are ASCII files written in a format that can be directly imported into Arc® GIS products for display. These files can also be easily converted to a format suitable for plotting with Generic Mapping Tools (GMT) with the command:

```
gmt grdconvert DepositFile___final.dat=ef out.grd
```

These files contain a six-line header followed by a 2-D matrix containing values of, for example deposit thickness. The header looks like:

```
NCOLS 140
NROWS 140
XLLCORNER -2616390.
YLLCORNER 1742330.
CELLSIZE 5000.000 5000.000
NODATA_VALUE -9999.
```

The parameters `XLLCORNER` and `YLLCORNER` give the coordinates of the lower-left corner of the model domain in meters (if it is a projected grid) or degrees (if the grid is latitude/longitude). The cell size is in the same units as the corner coordinates. Ash3d can generate grids whose x and y spacing are unequal, and this is typically the case for simulations generated using the web interface. But Arc products require the x and y spacing to

Line	Output type	Variable	units	File names
1	ASCII	final deposit thickness	mm	DepositFile____final.dat
2	KML	final deposit thickness	mm	deposit_thickness_mm.kml
			inches	deposit_thickness_inches.kml
3	ASCII	transient deposit	mm	Deposit_xxx.xxhrs.dat
4	KML	transient deposit	mm	deposit_thickness_mm.kml
			inches	deposit_thickness_inches.kml
5	ASCII	ash-cloud concentration	mg/m^3	CloudConcentration_xxx.xxhrs.dat
6	KML	ash-cloud concentration	mg/m^3	CloudConcentration.kml
7	ASCII	ash-cloud height	km	CloudHeight_xxx.xxhrs.dat
8	KML	ash-cloud height	km	CloudHeight.kml
9	ASCII	transient ash-cloud load	T/km^2	CloudLoad_xxx.xxhrs.dat
10	KML	transient ash-cloud load	T/km^2	CloudLoad.kml
11	ASCII	deposit arrival times	hours	DepositArrivalTime.dat
12	KML	deposit arrival times	hours	ashfall_arrivaltimes_hours.kml
13	ASCII	cloud arrival times	hours	CloudArrivalTime.dat
14	KML	cloud arrival times	hours	cloud_arrivaltimes_hours.kml
15		consolidated output file		specified in input file

Table 3.3: Block 4 options for output data

be equal. The KML files are written using Keyhole Markup Language and can be opened by Google Earth® or other virtual globe software. The KML files can be zipped using the following Linux command, which reduces file size by about 90-95% and creates a KMZ file that can also be opened in Google Earth: `zip -r CloudLoad.kmz CloudLoad.kml`

The consolidated output file option specified in Line 15 of Table 3.3 serves several purposes. It is designed to store the store the full ash concentration for each grain-size bin of all cells at specified time steps in a format specified in Line 16. Current options for this output format is `ascii`, `binary`, or `netcdf`. If `netcdf` is specified, this output file can serve as a restart file, allowing a failed or suspended job to continue from the last output step. The NetCDF file contains the full contents of the input file used to run the job to facilitate its role as a restart file. The NetCDF file also contains the various output variables that can be selected in Lines 1-14 above. These include the following 2-D variables listed in Table **??**.

Optionally, an additional integer specifier can be included on Line 15 to indicate if this consolidated output file should only contain the 2-D variables, which dramatically reduces the size of the output file, at the expense of losing restart capabilities. If this specifier is either absent or is 1, then the 3-D concentrations are written. If it is 2, then only the 2-D output products are written.

Specifying `netcdf` in Line 16 provides the most general output product for post processing. If Line 16 contains `ascii`, then data are written in ASCII format to files names `3d_tephra_fall_xxx.dat` where `xxx` is the output hour marker. The output data are in column format in the following order: x (or lon), y (or lat), z, total concentration (in kg/km^3). If `binary` is specified, then then total ash concentration is written to the files

Variable Name	Dimensions	units
area	x,y	km^2
depothick	x,y,t	mm
depothickFin	x,y	mm
depotime	x,y	hours
ash_arrival_time	x,y	hours
ashcon_max	x,y,t	mg/m^3
cloud_height	x,y,t	km
cloud_load	x,y,t	T/km^3
radar_reflectivity	x,y,z,t	dB
cloud_bottom	x,y,t	km
depocon	x,y,gs,t	kg/m^2
ashcon	x,y,z,gs,t	kg/km^3

Table 3.4: Output Products

`3d_tephra_fall_xxx.raw` where `xxx` is the output hour marker. Similarly, the deposit thickness is written to the files `2d_tephra_depo_xxx.raw`. The following code sections shows how the data are written:

```
open(unit=20,file='3d_tephra_fall_'//cio//'.raw', &
     status='replace', &
     access='direct',recl=4*nxmax*nymax*nzmax)
write(20,rec=1)(((ashcon(i,j,k),i=1,nxmax),j=1,nymax),k=1,nzmax)
close(20)
open(unit=21,file='2d_tephra_depo_'//cio//'.raw', &
     status='replace', &
     access='direct',recl=4*nxmax*nymax)
write(21,rec=1)((DepositThickness(i,j),i=1,nxmax),j=1,nymax)
close(21)
```

To read these files, you will need to know the values for `nxmax`, `nymax`, and `nzmax`. Although the default precision for calculations in Ash3d is `real*8`, the default output precision is `real*4` in order to reduce output file size. This can be changed, if desired by editing the code in module `precis_param` to set the internal precision (`ip`) and the output precision (`op`). See 4 for more details on editing these and other compile-time parameters.

The 3-D ash concentration file specified in Line 15 of Table 3.3 contains the full 3-D distribution of each grain size in the model domain at specified times. The following lines complete this block of input:

Line 17 gives `nWriteTimes`, the number of times data are to be written out to the files above. Data may be written out at uneven intervals (e.g. 0.2, 3, 3.4, and 12 hours after the eruption start) or at even intervals (e.g. every two hours, starting 2 hours after the eruption start).

Line 18 gives the times at which output is written: If `nWriteTimes`> 0, Line 18 should contain `nWriteTimes` numbers, in increasing order, which specify the times in hours after the start of the eruption at which the above data are to be written. If `nWriteTimes`$= -1$, Line 18 should contain a single number indicating the time interval in hours between write times.

Block 5 : Input wind files
This block should contain a number of lines equal to `nWindFiles` (Block 3, Line 5). For the case of `iwind`$= 5$, `nWindFiles` should be 1 and Block 5 Line 1 should be the path to the top-level directory of that product. For example, for the NCEP 2.5° Reanalysis files stored as:

```
/data/WindFiles/NCEP
|-- 2016
|   |-- air.2016.nc
|   |-- hgt.2016.nc
|   |-- omega.2016.nc
|   |-- uwnd.2016.nc
|   `-- vwnd.2016.nc
|-- 2017
    |-- air.2017.nc
    |-- hgt.2017.nc
    |-- omega.2017.nc
    |-- uwnd.2017.nc
    `-- vwnd.2017.nc
```

only `/data/WindFiles/NCEP` should be given on Line 1. Unless `iwindformat` (Block 3, Line 1) is 1 or 2, these files should be in either NetCDF or GRIB format and should consist of output from one of the model types listed in Table 3.2. Each line gives the file name and path. In the example file, Ash3d reads a single wind file named `latest.nc`, located in the subdirectory `Wind_nc`. Ash3d opens this file and does a preliminary read to ensure that it covers the time period and geographic region specified in input.

Block 6 : Arrival times at airports
The input lines in this block specify whether arrival times and other information at specific locations are to be written to output. Normally, Ash3d reads from a text input file containing the latitude and longitude (or x and y locations) of a list of points. For the web interface, this is a global list of airports, but when hand-editing the ASCII input file, other lists, such as sample locations, may also be specified. Ash3d can then generate either text or KML files listing the subset of these locations where ash was deposited or where the cloud passed overhead. Ash3d can also write out the grain-size distribution of points at those locations. All but line 4 require "yes" or "no" parameters.

Line 1 indicates whether to write out ash arrival times at these locations to an ASCII file. If "yes" is given, an output file is generated called `ash_arrivaltimes_airports.txt` in the format shown in Figure 3.4.

```
ARRIVAL TIME OF ASH IN AREA MODELED BY ASH3D
simulation using input file: ash3d_input_1.inp
Model run date: 2021.04.22, time UTC:     0:41

-----------------------------------------
        SOURCE PARAMETERS FOR SIMULATION OF: Crater Lake
Pulse      start time      Duration      Plume height      volume
             UTC             hrs          feet            km3
  1   1980-05-18T17:28:00Z    24.00      40.0  131232.     40.00000
-----------------------------------------
```

	LOCATION			CLOUD			DEPOSIT			
				Cloud Arrival Time		Duration	Deposit Arrival Time		Duration	Thickness
(Airport code &) Place name	Latitude	Longitude		hrs after			hrs after			
				date/time UTC	start	hrs	date/time UTC	start	hrs	mm NWS rank
DTA Delta Municipal Airport,	39.3792	-112.5083		1980-05-19T12:38:00Z	19.16	>20.78	1980-05-19T18:49:00Z	25.34	>14.61	0.38 trace or less
AMA Amarillo International Ai	35.2272	-101.7219		1980-05-19T21:32:00Z	28.06	>11.89	0000-00-00T00:00:00Z	-9999.00	0.00	0.00 trace or less
FIL Fillmore Airport, UT	38.9583	-112.3625		1980-05-19T13:43:00Z	20.24	>19.71	1980-05-19T22:25:00Z	28.94	>11.00	0.21 trace or less
FDR Frederick Municipal, OK	34.3522	-98.9847		1980-05-20T01:50:00Z	32.35	> 7.59	0000-00-00T00:00:00Z	-9999.00	0.00	0.00 trace or less
LVM Livingston, MT	45.6667	-110.5667		1980-05-20T02:49:00Z	33.35	> 6.59	0000-00-00T00:00:00Z	-9999.00	0.00	0.00 trace or less
ORl Grants Pass Airport, OR	42.5117	-123.3878		1980-05-19T11:27:00Z	17.98	19.79	0000-00-00T00:00:00Z	-9999.00	0.00	0.00 trace or less
JDA John Day, OR	44.4053	-118.9617		1980-05-18T19:37:00Z	2.15	>37.80	1980-05-18T21:44:00Z	4.27	>35.68	7.93 substantial
JZP Prospect State, OR	42.7431	-122.4881		1980-05-18T17:31:00Z	0.04	>39.91	1980-05-18T17:31:00Z	0.04	39.10	691.40 severe
PDT Pendleton Muni, OR	45.6983	-118.8361		1980-05-20T08:30:00Z	39.03	> 0.92	0000-00-00T00:00:00Z	-9999.00	0.00	0.00 trace or less
ONO Ontario Muni, OR	44.0228	-117.0139		1980-05-18T20:49:00Z	3.35	>36.60	1980-05-18T23:44:00Z	6.27	28.92	56.71 heavy
VWB Myrtle Creek Muni, OR	42.9983	-123.3094		1980-05-19T11:15:00Z	17.78	6.33	0000-00-00T00:00:00Z	-9999.00	0.00	0.00 trace or less
MFR Medford Jackson Co, OR	42.3736	-122.8722		1980-05-19T10:25:00Z	16.94	>23.01	1980-05-19T15:24:00Z	21.93	14.65	0.11 trace or less
MDJ City County, OR	44.6661	-121.1631		1980-05-20T06:44:00Z	37.26	> 2.68	0000-00-00T00:00:00Z	-9999.00	0.00	0.00 trace or less
LKV Lake Co, OR	42.1611	-120.3981		1980-05-18T22:39:00Z	5.18	>34.76	1980-05-19T05:12:00Z	11.72	>28.22	2211.15 severe
LGD La Grande/Union Co, OR	45.2894	-118.0050		1980-05-19T10:39:00Z	17.17	>22.78	1980-05-19T14:06:00Z	20.62	0.00	0.04 trace or less
LMT Kingsley Field, OR	42.1631	-121.7358		1980-05-18T23:07:00Z	5.64	>34.30	1980-05-19T05:49:00Z	12.34	>27.61	417.55 severe
HES Hermiston State, OR	45.8258	-119.2583		1980-05-20T08:57:00Z	39.49	> 0.46	0000-00-00T00:00:00Z	-9999.00	0.00	0.00 trace or less
UXV Enterprise Muni, OR	45.4247	-117.2650		1980-05-19T10:25:00Z	16.94	>23.01	1980-05-19T16:14:00Z	22.77	0.00	0.03 trace or less
CHZ Chiloquin State, OR	42.5764	-121.8769		1980-05-18T17:31:00Z	0.04	>39.91	1980-05-18T20:02:00Z	2.57	>37.38	421.99 severe
BNO Burns Muni, OR	43.5906	-118.9542		1980-05-18T18:49:00Z	1.34	>38.60	1980-05-18T21:33:00Z	4.07	>35.87	505.60 severe
RDM Roberts Field, OR	44.2544	-121.1514		1980-05-18T18:35:00Z	1.11	>38.83	1980-05-18T22:03:00Z	4.57	>35.37	0.12 trace or less
BKE Baker Muni, OR	44.8383	-117.8100		1980-05-18T22:00:00Z	4.53	>35.41	1980-05-19T05:32:00Z	12.07	11.70	0.20 trace or less
XXU Ashland Muni-Sumner Parke	42.1911	-122.6589		1980-05-19T09:00:00Z	15.52	>24.42	1980-05-19T09:00:00Z	15.52	22.89	1.55 minor
XXN Ells Field-Willits Munici	39.4511	-123.3689		1980-05-20T08:07:00Z	38.64	> 1.30	0000-00-00T00:00:00Z	-9999.00	0.00	0.00 trace or less
VWM Weed, CA	41.4747	-122.4531		1980-05-19T14:31:00Z	21.04	>18.90	1980-05-19T19:19:00Z	25.84	14.11	0.67 trace or less
WQD Tulelake Municipal, CA	41.8875	-121.3583		1980-05-19T02:21:00Z	8.87	>31.07	1980-05-19T09:23:00Z	15.91	>24.04	576.67 severe
QQM Shingletown, CA	40.5222	-121.8167		1980-05-19T15:24:00Z	21.93	>18.02	0000-00-00T00:00:00Z	-9999.00	0.00	0.01 trace or less
RDD Redding Muni, CA	40.5083	-122.2922		1980-05-20T01:08:00Z	31.66	> 8.28	0000-00-00T00:00:00Z	-9999.00	0.00	0.00 trace or less
RBL Red Bluff Muni, CA	40.1514	-122.2511		1980-05-20T02:31:00Z	33.04	> 6.90	0000-00-00T00:00:00Z	-9999.00	0.00	0.00 trace or less
FVP Gansner Field, CA	39.9439	-120.9469		1980-05-20T02:08:00Z	32.66	> 7.29	0000-00-00T00:00:00Z	-9999.00	0.00	0.00 trace or less
ZXW Skypark, CA	39.7097	-121.6153		1980-05-20T03:31:00Z	34.04	> 5.90	0000-00-00T00:00:00Z	-9999.00	0.00	0.00 trace or less
SIY Siskiyou County, CA	41.7814	-122.4681		1980-05-19T13:13:00Z	19.74	>20.21	1980-05-19T14:03:00Z	20.58	>19.36	3.31 minor
QQK Garberville, CA	40.0861	-123.8125		1980-05-20T06:42:00Z	37.23	> 2.72	0000-00-00T00:00:00Z	-9999.00	0.00	0.00 trace or less
FOB Fort Bragg, CA	39.4744	-123.7944		1980-05-20T07:32:00Z	38.07	> 1.88	0000-00-00T00:00:00Z	-9999.00	0.00	0.00 trace or less
WQH Butte Valley, CA	41.8872	-121.9744		1980-05-19T03:54:00Z	10.42	>29.53	1980-05-19T10:41:00Z	17.21	>22.74	91.29 heavy
WQE Dinsmore Municipal, CA	40.4931	-123.5986		1980-05-20T06:42:00Z	37.23	> 2.72	0000-00-00T00:00:00Z	-9999.00	0.00	0.00 trace or less
XXM Round Valley, CA	39.7903	-123.2653		1980-05-20T05:17:00Z	35.81	> 4.14	0000-00-00T00:00:00Z	-9999.00	0.00	0.00 trace or less
CIC Chico Muni, CA	39.7958	-121.8567		1980-05-20T03:31:00Z	34.04	> 5.90	0000-00-00T00:00:00Z	-9999.00	0.00	0.00 trace or less
HNX Hanna, WY	41.8306	-106.5333		1980-05-19T19:09:00Z	25.68	13.69	0000-00-00T00:00:00Z	-9999.00	0.00	0.00 trace or less
WRL Worland Municipal, WY	43.9675	-107.9531		1980-05-20T01:08:00Z	31.66	> 8.28	0000-00-00T00:00:00Z	-9999.00	0.00	0.00 trace or less
THP Hot Sprgs Co Thermopolis	43.6583	-108.2133		1980-05-19T21:57:00Z	28.48	>11.46	0000-00-00T00:00:00Z	-9999.00	0.00	0.00 trace or less
SHR Sheridan County, WY	44.7742	-106.9772		1980-05-20T05:49:00Z	36.34	> 3.60	0000-00-00T00:00:00Z	-9999.00	0.00	0.00 trace or less
SAA Shively Field, WY	41.4450	-106.8100		1980-05-19T19:09:00Z	25.68	>14.26	0000-00-00T00:00:00Z	-9999.00	0.00	0.00 trace or less
RKS Rock Springs Sweetwater C	41.5964	-109.0658		1980-05-19T14:58:00Z	21.50	>18.44	1980-05-19T20:11:00Z	26.72	0.00	0.04 trace or less
RIW Riverton Muni, WY	43.0644	-108.4569		1980-05-19T16:17:00Z	22.81	>17.14	0000-00-00T00:00:00Z	-9999.00	0.00	0.01 trace or less
RWL Rawlins Municipal, WY	41.8044	-107.2014		1980-05-19T17:58:00Z	24.50	>15.45	0000-00-00T00:00:00Z	-9999.00	0.00	0.00 trace or less
POY Powell, WY	44.8703	-108.7914		1980-05-20T01:15:00Z	31.78	> 8.17	0000-00-00T00:00:00Z	-9999.00	0.00	0.00 trace or less
LAR General Brees Field, WY	41.3136	-105.6731		1980-05-19T21:18:00Z	27.83	10.97	0000-00-00T00:00:00Z	-9999.00	0.00	0.00 trace or less
EMM Kemmerer Muni, WY	41.8250	-110.5583		1980-05-19T09:39:00Z	16.17	>23.77	1980-05-19T15:03:00Z	21.58	10.05	0.27 trace or less

Figure 3.4: Example of output file **ash_arrivaltimes_airports.txt**

Line 2 indicates whether the ASCII output file should give a grain-size distribution of the tephra deposit at each location.

Line 3 indicates whether to write out a KML file showing the locations that will be impacted by ash. If "yes" is given, a file named `ash_arrivaltimes_airports.kml` is generated that plots each impacted location as a red placemarks as shown in Figure 3.1. Additionally, files for each impacted airport are generated that contain the data for the ash accumulation as a function of time. These files are named `depTS_xxxx.dat` (where `xxxx` is the airport number in order of time of ash arrival) and have two columns containing hour and ashfall thickness in mm. Additionally, a gnuplot script is written for each of these airport locations so that time-series plots of ashfall accumulation can be embedded in the KML files in post-processing.

Line 4 gives the name and path of the file containing airports or other locations of interest. The format of this file must contain four columns with the location information (latitude, longitude, x, y), followed by the 3-character IATA airport code, then a 42-character descriptive string such as location. While the IATA code is currently not used in any output products, the descriptive string is use in ash arrival output products. An example of the expected format for the airport file is given below.

```
#   Latitude   Longitude          x           y  Code Location
    41.60833   -88.09417     0.00000     0.00000  II2  Lewis University Airport, IL
    34.68861   -85.29056     0.00000     0.00000  GA1  Barwick Lafayette Airport, GA
    58.36528  -152.69667     0.00000     0.00000  A33  Hidden Lake, AK
```

In this case, airports with latitude and longitudes are given, but the x and y columns are filled with 0.0 as a place-holder. If the computational grid is Lon/Lat, then these airport coordinates will be used. If the computational grid is projected, then the x and y columns can be filled with the appropriate values for the coordinate system used. Alternatively, the internal projection routines can be used to generate the x and y values from the latitude and longitude columns. This would over-write any values read in from the x and y columns of the file. To specify that Ash3d should calculate the projected coordinates, Line 5 of this block can be set to "yes". Ash3d would then convert the latitude and longitude to the projected system specified in Block 1, Line 2. If this Line is "no" and if the computational grid is projected, then the x and y columns of the airport file are used and the latitude and longitude columns are ignored.

Included with Ash3d is an internal list of 6117 global airports with IATA codes, locations, and coordinates. By default, this is a filled variable at compile-time, but optionally can be read at run-time from the file `/opt/USGS/Ash3d/share/GlobalAirports_ewert.txt`. If Line 4 of this block is empty, or has the word "internal", then this built in list of global airports is used. If an airport file name is provided, then this internal list is replaced. Alternatively, if the first character of the file name provided is '+', then the contents of the file are appended to the internal global list. For example, if the block had the following form:

```
****************** BLOCK 6 ***************************************************
yes               # Write out ash arrival times at airports to ASCII FILE?
no                # Write out grain-size distribution to ASCII airport file?
yes               # Write out ash arrival times to kml file?
+PointOfInt.txt   # Name of file containing aiport/POI locations
yes               # Have libprojection calculate projected coordinates?
```

****

then the global airport list would be extended by the contents of PointOfInterest.txt with the x and y columns ignored in favor of the longitude and latitude or internally projected mappings. The maximum number of airports is currently set to 10,000.

Block 7 : Grain size groups

Ash3d can run an unlimited number of grain sizes, although the required run time increases roughly in proportion to the number of grain sizes. To reduce model run time, Ash3d stops calculations for individual grain sizes once they have left the simulation domain, either advected out the sides or deposited. A grain size bin is flagged as having left the simulation domain when 1 g or less remains of that grain size bin throughout the model domain. When simulating ash clouds, one very small grain size (0.01 mm for example) with a very low settling velocity, is frequently adequate to display general cloud movement. When simulating deposits, it is best to use at least a half dozen grain sizes: fewer grain sizes tend to produce secondary thickness maxima as an artifact [**?**].

Line 1 of Block 7 consists first of an integer giving the number of size bins (`nsize`). Optionally, a second integer can be provided which specifies the fall model to be used (`FV_ID`). If `FV_ID` is provided, a third optional integer can be provided specifying the shape factor used: 1 for F and G, or 2 for sphericity, ϕ.

`FV_ID`	Fall model	Expected parameters
N/A	Wilson/Huang	F [0.44]
0	Tracer only	N/A
1	Wilson/Huang	F [0.44]
2	Wilson/Huang with slip-flow	F [0.44]
3	Pfiffer mod. of Wilson/Huang	F [0.44]
4	Ganser	F, G [0.44, 1.0]
5	Ganser with slip-flow	F, G [0.44, 1.0]
6	Stokes flow with slip-flow	N/A

Table 3.5: Particle fall models

If `FV_ID` is not given or `FV_ID`=1, the Wilson and Huang [23] model is used. If `FV_ID=2`, the Wilson and Huang model with a slip-flow correction is used [20]. If `FV_ID=3`, the modification to the Wilson and Huang model outlined by [18] is used. If `FV_ID`=4, the model of [7] is used and if 6, then the Ganser model with the slip-flow correction. If `FV_ID`=6, Stokes flow for spherical particles with a slip-flow correction is used. Optionally, fall velocities can be deactivated by setting `FV_ID`=0. In 3.5, the default values for F and G are shown in bracket which Ash3d uses if the corresponding columns of Block 7 are absent.

The first line of this block is followed by `nsize` lines, one for each bin size. These lines may contains two, three, four or five numerical parameters:

If two parameters: Ash3d reads the first as the mass fraction of that size bin, and the second as the settling velocity in m/s. Ash3d uses this as a constant settling velocity, inde-

pendent of elevation or fall velocity model. For example, the following block specifies two grain classes, 70% with a fall velocity of 0.01 m/s and 30% with a fall velocity of 1.0 m/s.

```
**********************************************************************************
2                              # Minimalist Grain-size specification
1.00 0.3
0.01 0.7
**********************************************************************************
```

If three parameters: Ash3d reads the first as grain size in millimeters, the second as the mass fraction, and the third as the density in kg/m^3. Ash3d calculates settling velocity of these particles assuming a shape factor (F) of 0.44, which is the average of values measured by Wilson and Huang [23]. The settling velocity in this case depends on air density and viscosity. Air density is calculated from the pressure and temperature at a given elevation (obtained from the NWP model data). The viscosity is calculated from Sutherlands Law [9], p. 201. The example below shows the grain-size distribution used for the airborne runs on the Ash3d webserver.

```
**********************************************************************************
1                              # Web server airborne case
0.0100 1.00 2000.
**********************************************************************************
```

If four parameters: Ash3d reads the first three as before, and the last as a shape factor. For the Wilson and Huang class of models (FV_ID=1,2, or 3) the shape factor, F, is defined as $F = (b+c)/2a$, where a, b, and c are the semi-major, intermediate, and semi-minor diameters of an ellipsoid.

```
**********************************************************************************
14 2                           # McGimsey Spurr
4       0.07303974221267455 800.0  0.8
2       0.07303974221267455 800.0  0.8
1       0.05907626208378088 800.0  0.8
0.5     0.04403866809881848 800.0  0.8
0.25    0.06337271750805586 1083.0 0.8
0.125   0.25026852846401720 1790.6 0.8
0.0625  0.12567132116004298 2000.0 0.8
0.03125 0.11815252416756176 2000.0 0.8
0.01563 0.08700322234156821 2000.0 0.8
0.00781 0.05692803437164339 2000.0 0.8
0.0039  0.03114930182599356 2000.0 0.8
0.00195 0.01396348012889366 2000.0 0.8
0.00098 0.00322234156820623 2000.0 0.8
0.00049 0.00107411385606874 2000.0 0.8
**********************************************************************************
```

For the Ganser model, shape is characterized by the sphericity, ϕ, defined as the ratio of the surface area of a sphere with equivalent volume to the actual surface area of the particle. If four parameters are provided with the Ganser model specified, then Ash3d interprets the fourth term as the Wilson and Huang shape parameter F, but additionally assumes $b = c < a$ (i.e. only prolate ellipsoids) so as to calculate area and volume of the particles and thereby ϕ. If a fifth parameter is given, then it is interpreted to be the ratio c/b, which allows oblate ellipsoids to be specified.

```
**************************************************************************
4 4                          # Ganser
0.125   0.2 1790.6 0.8 1.0
0.0625  0.2 2000.0 0.8 1.0
0.03125 0.3 2000.0 0.3 1.0  # needles
0.03125 0.3 2000.0 0.9 0.2  # flakes
**************************************************************************
```

Alternatively, sphericity can be directly specified using `Shape_ID`=2.

```
**************************************************************************
4 4 2                        # Ganser
0.125   0.2 1790.6 0.8       #  sphericity = 0.8 = Area_vol_equl_sphere/Area_part
0.0625  0.2 2000.0 0.8
0.03125 0.3 2000.0 0.5
0.03125 0.3 2000.0 0.6
**************************************************************************
```

See D for further discussion on the implementation and subtleties of the various fall models.

The grain-size specifications can be used to describe multiple populations of particles if desired. For example, the default grain-size distribution used on the Ash3d web-server for deposit simulations contains 6 bins (1mm $\Rightarrow$ 88μm) describing the primary grain-size distribution but also includes 4 bins to describe a distribution of aggregates with a lower density and more equant shape.

```
**************************************************************************
12                            # Web-server dep
2         0.06118 800     0.44
1         0.07098 1040    0.44
0.5       0.22701 1280    0.44
0.25      0.21868 1520    0.44
0.1768    0.05362 1640    0.44
0.125     0.04039 1760    0.44
0.088     0.02814 1880    0.44
0.2176    0.018   600     1.0
0.2031    0.072   600     1.0
0.1895    0.12    600     1.0
0.1768    0.072   600     1.0
0.1649    0.018   600     1.0
**************************************************************************
```

If the diameter of the last grain-size bin is negative, then the last grain-size line is reinterpreted to allow a log-normal grain-size distribution. In this case, the commonly-used logarithmic specification of grain-size, $\phi = -\log_2 d$ (with d in mm), is used where ϕ is normally distributed. The second and third values of this last line are interpreted to be the mean (μ_ϕ) and standard deviation (σ_ϕ). The remaining mass fraction not accounted for in the previously specified bins is distributed among all the bin according to

$$N = \frac{1}{\sigma_\phi \sqrt{2\pi}} \exp\left[-\frac{1}{2}\left(\frac{\phi - \mu_\phi}{\sigma_\phi}\right)^2\right] \tag{3.1}$$

Note that using the log-normal distribution assumes that all the grain-size bins are part of the primary grain-size distribution with no bins representing special classes such as aggregates. Secondly, Ash3d assumes that the bins are listed in order from smallest to largest and will sort all grain sizes to ensure this order. This sorting would interfere with multiple grain populations if an aggregate population or a flake/needle distinction is specified. The example below shows a log-normal grain-size distribution with a mean of $\mu_\phi = 5.5$ and a standard deviation of $\sigma_\phi = 2$. Note that this grain-size order will be reversed.

```
**************************************************************************
15 2                          # Gaussian
4       0.000 800.0  0.8
2       0.025 800.0  0.8
1       0.100 800.0  0.8
0.5     0.025 800.0  0.8
0.25    0.0 1083.0 0.8
0.125   0.0 1790.6 0.8
0.0625  0.0 2000.0 0.8
0.03125 0.0 2000.0 0.8
0.01563 0.0 2000.0 0.8
0.00781 0.0 2000.0 0.8
0.0039  0.0 2000.0 0.8
0.00195 0.0 2000.0 0.8
0.00098 0.0 2000.0 0.8
0.00049 0.0 2000.0 0.8
-1 5.5 2
**************************************************************************
```

Several of the above example grain-size distributions are shown in Figure 3.5 illustrating how this feature can be used to implement bimodal distributions.

Block 8 : Locations of vertical profiles

In some cases it has been useful to plot vertical profiles through the ash cloud, as for example, when ground-based LiDAR data were available over Europe during the 2010 eruption of Eyjafjallajökull. In that case, the lines in Block 8 appeared as follows:

The first line contains an integer indicating the number of locations (`nlocs`) where vertical profiles are to be recorded. If `nlocs`=0 (as in the example input file), no lines follow in this block. Otherwise, `nlocs` lines follow, giving the locations for each profile in the coordinate system appropriate for this model run (in this case, degrees longitude and latitude, respectively), followed by an optional station name. Upon execution, Ash3d writes out text files with the names v`vprofile0001.txt`, `vprofile0002.txt`, `vprofile0003.txt`, `vprofile0004.txt`, containing vertical profiles at each of these locations. Below is a truncated illustration of the output contained in `vprofile0001.txt`:

The header gives the x and y coordinates of this location. The numbers in the first row of the table, “0.250”, “0.750”, “1.250”, etc., are the elevations (km) of each cell center in the model. Each row of data following contains the date and time (UTC, in yyyymmddhh.hh), the number of hours after the beginning of the eruption, and the concentrations (mg m$^-$3) of tephra at each elevation. Ash3d writes out a line of output every 10 time steps as long as the cloud is passing overhead.

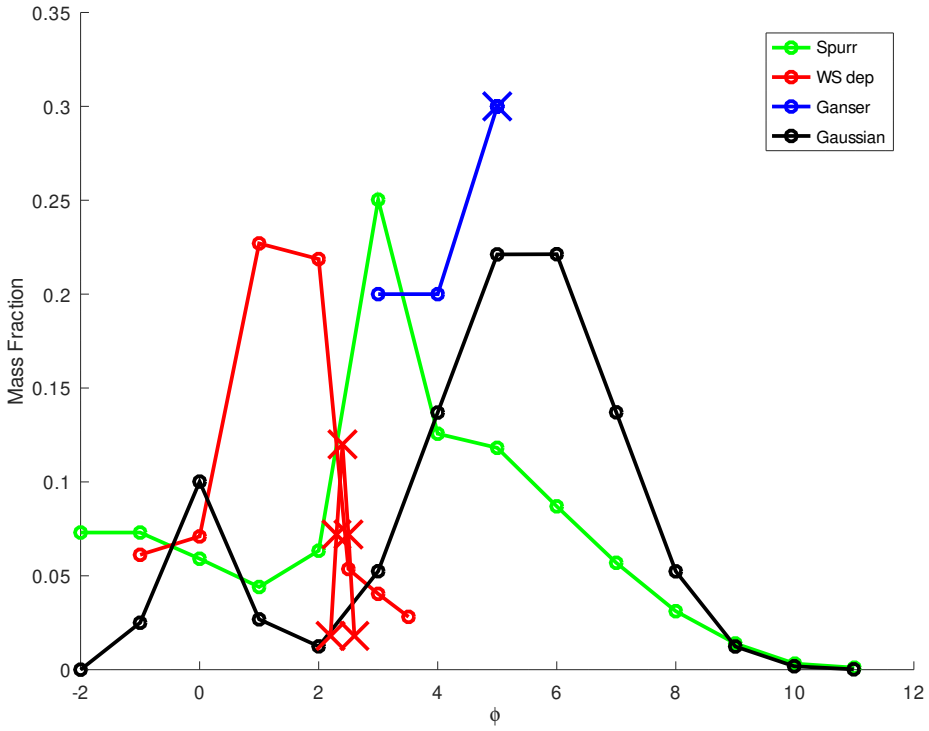


Figure 3.5: Example grain-size distributions. The secondary populations (aggregates for the web server deposit simulations and the flakes for the Ganser model) are shown with crosses

Block 9 : Comment lines
Line 1 of this block gives the name of the output file containing the 3-D cloud structure, if this output is specified in Line 15 of Block 4. Line 2 gives the title of the simulation. If the 3-D cloud structure is written out to a NetCDF file (as specified in Line 16 of Block 4), this title is included in that file. Line 3 gives an optional comment that may be written out to the same NetCDF file.

3.2.2 Extensions to the control file: Optional Modules

After these nine required blocks have been read, Ash3d scans the input file to search for the term "`OPTMOD=`". If this is found, then the corresponding optional module input block is read. This is primarily used in the development of research branches of the code, but there are a few that are part of the main code.

`RESETPARAMS` If a block with `OPTMOD=RESETPARAMS` is found, then a block of the input file is read that allows the user to reset the default value of several global variables. For example, the following block would reset the default magma density and deposit density.

```
**********************
# Reset parameters
**********************
OPTMOD=RESETPARAMS
 MagmaDensity   = 3500.0
 DepositDensity = 1300.0
********************************************************************************
```

The full list of parameters that can be reset are given in Table 3.6.

3.2.3 Environment variables

Some variables can be set at run-time through the use of environment variables on a Unix system. These are not needed, but can be used to change the default values if desired. There are four environment variables that Ash3d will test at run-time:

`ASH3DHOME:` If the Ash3d executable was compiled to read the airport and volcano database files at run-time, the path to these external files should be set when compiled. The default is `/opt/USGS/Ash3d`, but this is set in the `makefile` by the `INSTALLDIR` variable. If the files were subsequently moved, or if different versions were needed, Ash3d tests for the environment variable

`ASH3DCFL:` The CFL factor is set by default to 0.8, but can be overwritten by the environment variable `ASH3DCFL`. Note that this factor could be subsequently reset if the optional input block

name	Default value	Units	Description
MagmaDensity	2500.0	kg/m^3	Density for DRE calculations
DepositDensity	1000.0	kg/m^3	Density of tephra deposit
LAM_GS_THRESH	250.0	m/	mean free path for Cunningham slip
AIRBORNE_THRESH	1.0e-3	kg	Tot. Airborne mass threshold for bin skipping
GRAV	9.81	m/s^2	Gravitational acceleration
RAD_EARTH	6371.229	km	Radius of Earth
CFL	0.80	unitless	Courant-Friedrichs-Lewy constant
DT_MIN	1.0e-5	hour	Minimum allowed time-step
DT_MAX	1.0	hour	Maximum allowed time-step
ZPADDING	1.3	unitless	Scaling factor for height of z-grid
ZSCALING_ID	0	case ID	z-grid case (0=No topo;1=shifted;2=scaled)
DEPO_THRESH	1.0e-1	mm	Threshold value to output deposit
DEPRATE_THRESH	1.0e-2	mm/hour	Threshold value to output deposit rate
CLOUDCON_THRESH	1.0e-3	t/km^3	Threshold value to output cloud concentration
CLOUDCON_GRID_THRESH	1.0e-7	t/km^3	Threshold value for FAST_SUBGRID
CLOUDLOAD_THRESH	1.0e-2	t/km^2	Threshold value to output cloud load
THICKNESS_THRESH	1.0e-3	mm	Threshold value to output deposit
StopValue_FracAshDep	0.99	unitless	Stop condition for fraction deposited
DBZ_THRESH	-2.0e+1	dB	Threshold value to output radar reflectivity
VelMod_umb	1	case ID	Umbrella velocity model: 1 [4] or 2 [22]
lambda_umb	0.2	unitless	Suzuki constant for umbrella cloud
N_BV_umb	0.02	s^{-1}	Brunt-Vaïsala frequency for umbrella cloud
k_entrainment_umb	0.1	unitless	Entrainment coefficient for umbrella cloud
SuzK_umb	12.0	unitless	Suzuki parameter for umbrella cloud
useMoistureVars	0	logical	0/1=F/T use RelHum and SpecHum
useVz_rhoG	1	logical	0/1=F/T Vz=FinDif v.s. Vz=rho.grav
useWindVars	0	logical	0/1=F/T include vel in output
useOutprodVars	1	logical	0/1=F/T include 2-D products in output
useRestartVars	0	logical	0/1=F/T include ashcon in output
cdf_institution	USGS	text	Institution name
cdf_run_class	Analysis	text	Analysis, Hypothetical, Forecast
cdf_url	url	text	Reference web page

Table 3.6: Parameters that can be reset through OPTMOD=RESETPARAMS

`ASH3DVERB:` When Ash3d is executing, it will write to standard output various notes about what it is doing, how it has interpreted the control file, information on the state of the run and progress as well as any closing notes. These notes are mirrored to a log file (`Ash3d.lst`). The user has some control over how verbose this output is by setting the verbosity environment variable `ASH3DVERB`. There are ten output levels with lower numbers indicating more messages to standard output. Levels are described in Table 3.7. The default level is 3. This can be over-ridden, if for example, there is an error and you would like further information about the position in the code causing the error, but running:

```
ASH3DVERB=2 ./Ash3d control.inp
```

Similarly, if you are running many Ash3d instances and do not want anything written to the screen, but do want the individual log files, you can use:

```
ASH3DVERB=9 ./Ash3d control.inp
```

Level 10 suppresses all output.

level	label	Description
1	debug2	Additional debugging information only written to stdout
2	debug1	Debugging information only written to stdout
3	log	Time step information (this is the limit for writing to logfile)
4	info	Additional information on run set up and shutdown
5	statistics	Details on health of run (timing, mass conservation)
6	production	Major program flow info
7	essential	Only start up and shutdown messages
8	error	No logging to stdout, only stderr (and logfile)
9	silent	No logging to stdout,stderr. Logfile written as normal
10	dark	No logging to stdout,stderr or logfile

Table 3.7: Ash3d verbosity levels

`ASH3DPLOT:` The post-processing utility, `Ash3d_PostProc` (described below in Section 3.3) can use several graphics packages for plotting maps and figures. There is a preference hierarchy built in to the code, based on output type (map, shapefile, vertical transect) as well as availability on the system. These can be over-ridden by setting the environment variable `ASH3DPLOT`=1-4, where 1 corresponds to DISLIN, 2 PLplot, 3 gnuplot, and 4 GMT. For example, the default contouring package for building shapefiles is gnuplot, but DISLIN can be used via:

```
ASH3DPLOT=1 ./Ash3d_PostProc 3d_tephra_fall.nc 5 5
```

`OMP_NUM_THREADS:` If Ash3d was compiled with OpenMP, then the number of threads available at run-time can be adjusted with this environment variable.

```
OMP_NUM_THREADS=4 ./Ash3d control.inp
```

3.2.4 Executing Ash3d

To run Ash3d, simply type `Ash3d` if the executable is in your path, or the full path to the executable (e.g. `/opt/USGS/Ash3d/bin/Ash3d`. This with start an interactive session where the user is prompted for control file. Next, the user will be prompted whether or not to load a concentration file. Normally, this is not needed and the user can enter `no` to continue with the Ash3d simulation. If `yes` is entered, this will allow users to continue a simulation that was aborted or ended prematurely.

```
[user@ash3d MtStHelens]$ ./Ash3d
 Enter name of ESP input file:
MSH.inp
 Load concentration file?
yes
 Enter name of concentration file
3d_tephra_fall.nc
   Step :  time
           1  0.250000000
           2  0.750000000
           3   1.25000000
           4   1.75000000
           5   2.25000000
           6   2.75000000
 Enter timestep for initialization
5
```

To run Ash3d non-interactively, simply provide the name of the control file as a command-line argument. This will begin the execution with messages and program progress written to standard output. All informational messages as well as error messages are mirrored to a log file (`Ash3d.lst`). As noted above, the verbosity of these messages can be adjusted with the environment variable `ASH3DVERB`. Also written during execution is the file `progress.txt`, which contains the fractional progress of the time stepping from start to the maximum anticipated time step. Simulations might complete prior to reaching 1.0 for a variety of reasons (all ash has deposited, anticipated number of time steps is greater than necessary, etc.).

In the initial stage of execution, Ash3d evaluates the control file, checks for any error in the computational grid layout, grain-size specification, etc.x. It also evaluates the files specifying the atmospheric conditions and checks for consistency with the computational grid and requested times. Ash3d then proceeds with the time integration of Equation 1.5. Stop conditions are evaluated each time step including: erupted mass has deposited ($< 1\%$ remaining aloft), time exceeds allotted time, all individual grain sizes have been flagged as deposited. Additional error checks are evaluated such as mass conservation errors greater than 0.001 or for any negative tephra volumes (volume aloft, deposited outflow, source). When a stop condition is met, finalizes output files and terminates.

3.3 Post-processing

There are many output files that Ash3d can generate as it is running; including ESRI ASCII files of the deposit accumulation, final deposit thickness, deposit arrival time, various measures of the airborne ash cloud (height, cloud load, maximum cloud concentration, cloud arrival time). These ASCII files can be loaded into GIS software such as Arc View or QGIS. Additionally, Ash3d can directly write KML files for each of the variable listed above which can be viewed interactively with virtual globe software such as Google Earth. Whether or not to write this output files is specified in Block 4 of the control file used for the Ash3d simulation. An example of a KML file for a final deposit is shown in Figure 3.6.

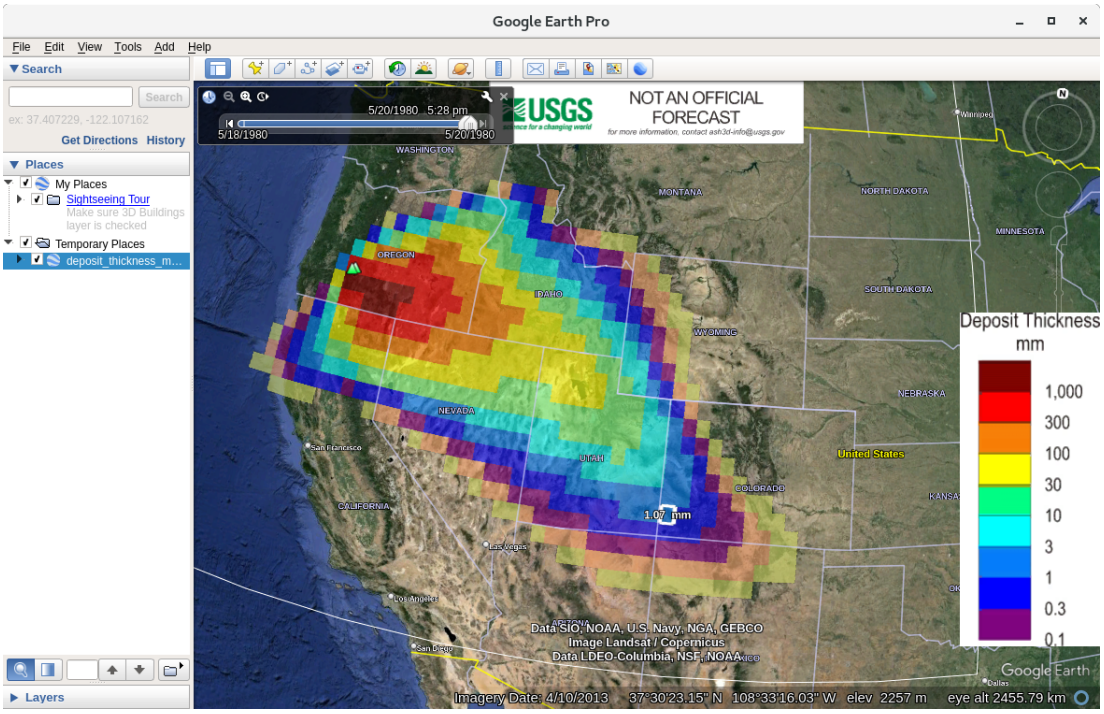


Figure 3.6: Viewing deposit KML output with Google-Earth

In the KML files, the domain of the Ash3d simulation is outlined with a white line and the cells are colored according to the cell-average value. The spatial resolution of the simulation is apparent, as there is no interpolation. Values of individual cells can be seen simply by mousing over the cell. KML files of deposit variables have cells clipped to the ground surface. Cloud variables are pinned to cloud height.

To view the ESRI ASCII files in GIS software, the files can be added as a raster layer on top of a basemap. The data might need to be shifted to the same periodic mapping of longitude. For example, the ASCII cloud height data for the example in Figure 3.6 has the following header:

```
NCOLS     76
NROWS     40
XLLCORNER          225.000
```

```
YLLCORNER          33.000
CELLSIZE           0.500          0.500
```

indicating a lower-left corner at a longitude of 225° E. The world coastline data has a range of $-180 \rightarrow 180$. The raster data must be shifted either within the viewing software or by editing the longitude. Figure 3.7 shows an example of Cloud Height viewed with QGIS.

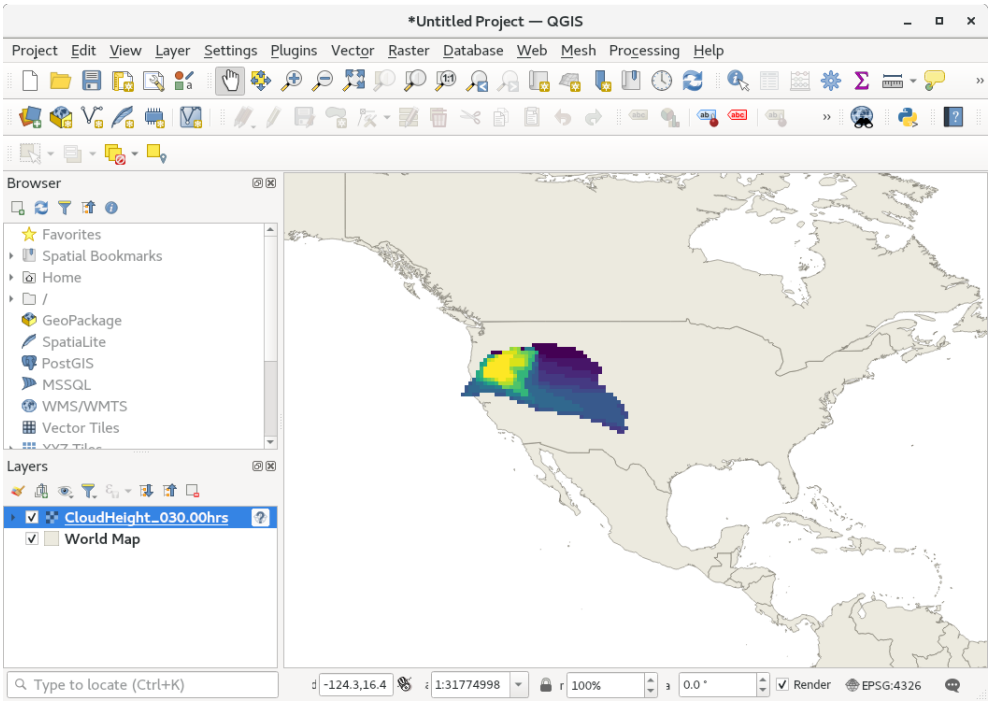


Figure 3.7: Viewing cloud height output with QGIS

Other output products written at run-time include text or KML files of ash arrival time at airports or points-of-interest (POI) as specified in Block 6 of the control file. This can also be viewed in Google Earth with the data appearing as white dots for locations that had ashfall and red dots for points only effected by the drifting cloud. Clicking on these points will open a descriptive window that gives the time of arrival of the cloud as well as duration. If the point is effected by ashfall, a time-series plot will be included showing the accumulation of the deposit at the site. These plots are generated by executing `gnuplot` as a system call within Ash3d, then executing `gzip` to bundle the KML file with the linked deposit plots into a compressed KMZ file. This information is summarized in the file `ash_arrivaltimes_airports.txt` (Figure 3.4).

Vertical profiles can be written in ASCII format, if requested in Block 8 and will typically require post-processing for visualization.

The most general and complete output is the 3-D ash concentration that can be requested on Line 15 of Block 4 of the control file. If `ASCII` is selected on Line 16, then a 3-D ESRI ASCII file of the total ash concentration is written which can be loaded into a GIS program (e.g. ArcView or QGIS). If `binary` is specified, then a binary ash concentration file can be

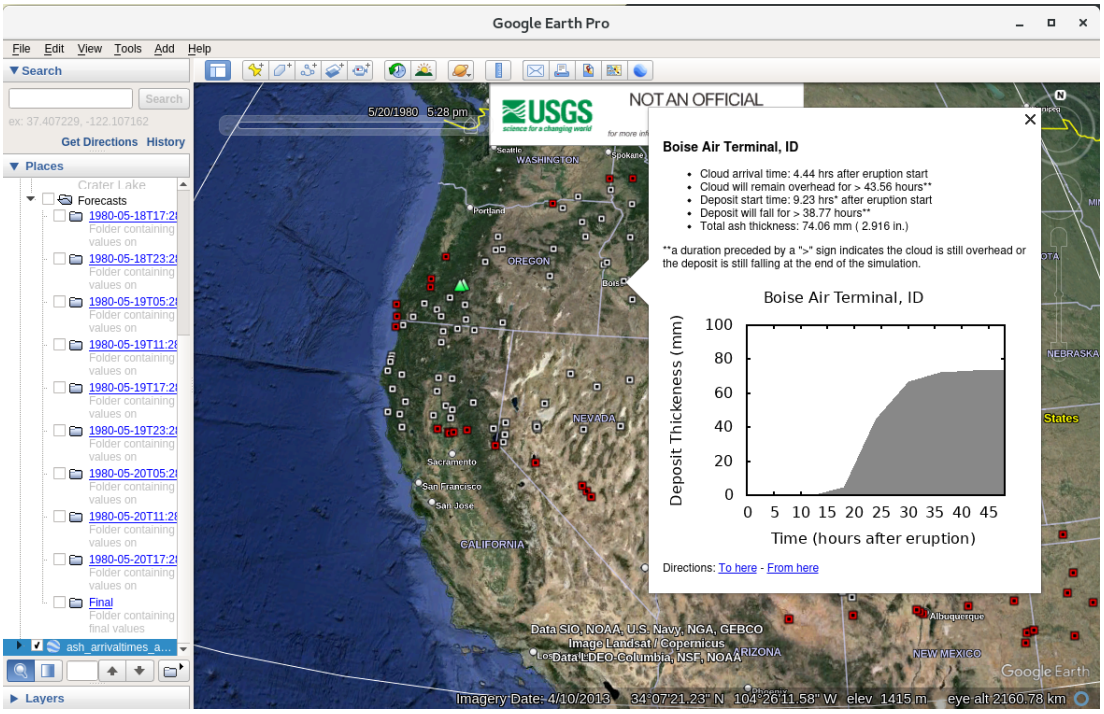


Figure 3.8: Viewing Airport arrival time KML output with Google-Earth

read into a more general visualization software such as ParaView. The recommended output format is `netcdf`. If this is selected, the full ash concentration data ($q(x, y, z, gs, t)$) and full deposit-concentration data is saved at the requested time steps. Additionally, all the derived variables listed above, such as deposit thickness, cloud height, arrival times, etc. are included. If Airport/POI data are requested in Block 4, then these data are also included in the NetCDF output file on the time steps requested in Block 4. If vertical profile data are requested in Block 8, then these data are also stored in the NetCDF file on the time-raster of that dataset (native time steps of the computation).

In addition to the ash concentration and derived variables listed above, all the information from the control file is included either as additional variables (grain-size values or eruptive pulse values) or as global attributes such as the values used for all resettable parameters, git commit IDs for Ash3d and MetReader, etc. This enables a complete reconstruction of the simulation from the information in the NetCDF file. Since this is such a convenient format and since the full ash concentration data needed for restarting simulations is seldom needed, cloud and deposit concentration variables can be suppressed if the optional second term on Line 15 of Block 4 is set to 2 (default is 1, indicating to write out restart concentrations).

```
yes 2   #Write out 3-D ash concentration at specified times?
```

To simply inspect the output, the NetCDF file is COARDS-compliant (naming convention for dimensions) and can be directly read by many visualization programs (e.g. ncview, shown in Figure 3.9). These programs display the data over a map and allow an easy visualization

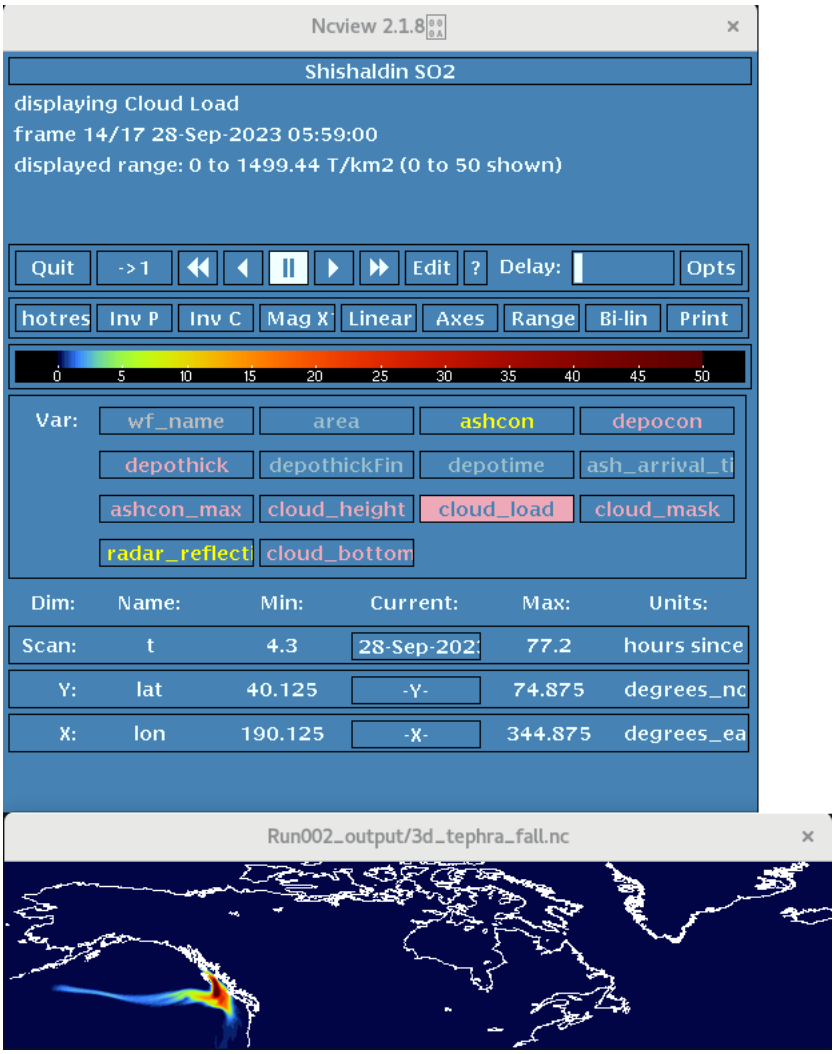


Figure 3.9: Displaying NetCDF file using general data viewer

of the evolution of the cloud or deposit.

3.3.1 Ash3d_PostProc

The Ash3d repository includes a post-processing tool (`Ash3d_PostProc`) that can be used to generate simple maps from the NetCDF output file. Any of the standard output products (ESRI ASCII, KML/KMZ, binary) can be recreated using this tool along with png images of 2-D data in mapview (or contour plots for vertical profiles) or shapefiles of 2-D data. To produce image files, `Ash3d_PostProc` can use a variety of geographic plotting software. Both DISLIN and PLplot are available as libraries and can be linked to `Ash3d_PostProc` at compilation time. DISLIN includes the feature where contour data can be accessed by `Ash3d_PostProc` and used to generate shapefiles. This is the preferred means of generating shapefiles on a Microsoft Windows system. `Ash3d_PostProc` can also generate maps using `gnuplot` and Generic Mapping Tools (GMT) via the writing and execution of temporary scripts.

`Ash3d_PostProc` is designed to be a quick tool for converting the NetCDF output of an Ash3d run to format that can be visualized. This tool can be run interactively by just running the executable with no command-line arguments. If one argument is provided, it is interpreted to be the name of a control file (described below). Minimal instructions for running this tool are available by running the program with a -h as the only argument

```
[user@ash3d example_problem]$ ./Ash3d_PostProc -h
 Dislin   T
 Plplot   T
 Gnuplot  T
 GMT      T

 Ash3d post-processing tool: Ash3d_PostProc

Usage: Ash3d_PostProc control_file [t_index]
           or
       Ash3d_PostProc infile output_product format
  where: infile   = the NetCDF file written by Ash3d
   output_product = 1 full concentration array
                    2 deposit granularity
                    3 deposit thickness (mm time-series)
                    4 deposit thickness (inches time-series)
                    5 deposit thickness (mm final)
                    6 deposit thickness (inches final)
                    7 ashfall arrival time (hours)
                    8 ashfall arrival at airports/POI (mm)
                    9 ash-cloud concentration (mg/m3)
                   10 ash-cloud height (km)
                   11 ash-cloud bottom (km)
                   12 ash-cloud load (T/km2 or )
                   13 ash-cloud radar reflectivity (dBz)
                   14 ash-cloud arrival time (hours)
                   15 topography
```

```
            16 profile plots
    format = 1 ASCII/ArcGIS
             2 KML/KMZ
             3 image/png
             4 binary
             5 shape file
[t_index] = index of time slice to plot; -1 for final (optional)
```

For example, to generate a contour plot of the final deposit thickness in mm from the output NetCDF file, enter:

```
Ash3d_PostProc 3d_tephra_fall.nc 5 3
```

The preferred graphics package is set at the time of compilation depending on the system (Linux, Windows, MacOS) and the availability of the libraries. These can always be overwitten at run-time with the environment variable `ASH3DPLOT`, where: 1=DISLIN, 2=PLplot, 3=gnuplot, and 4=GMT. Examples of these plotting packages is shown in Figure 3.10.

```
ASH3DPLOT=3 Ash3d_PostProc 3d_tephra_fall.nc 5 3
```

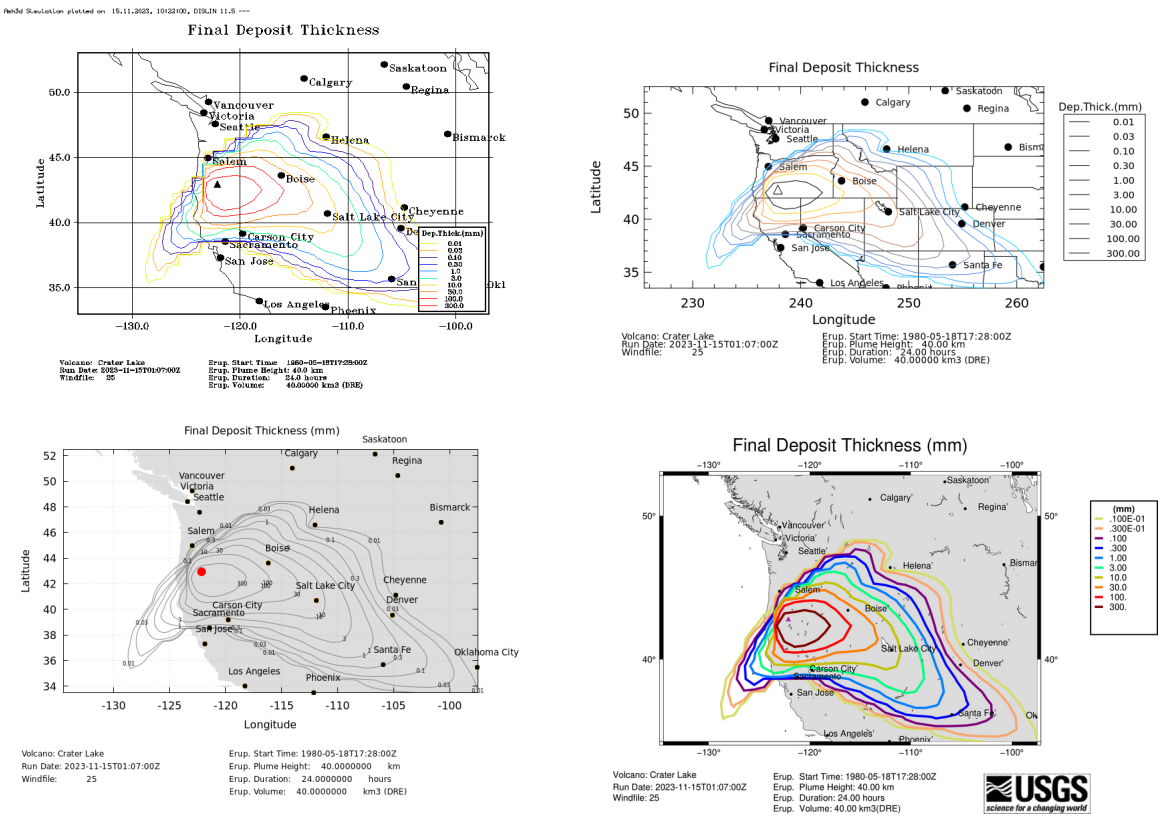


Figure 3.10: Final deposit contour maps using `Ash3d_PostProc` with different plotting packages.

To generate a shapefile of the cloud load at step 3 of the output file, enter:

```
Ash3d_PostProc 3d_tephra_fall.nc 12 5 3
```

This will create four files: `AshCdLod.shx`, `AshCdLod.shp`, `AshCdLod.prj`, `AshCdLod.dbf`, which are then bundled into `AshCdLod.zip` using a system call to `gzip`. This shapefile can then be loaded as vector layer in a GIS program. Figure 3.11 shows the shapefile generated above using QGIS and Figure 3.12 shows the shapefile attributes.

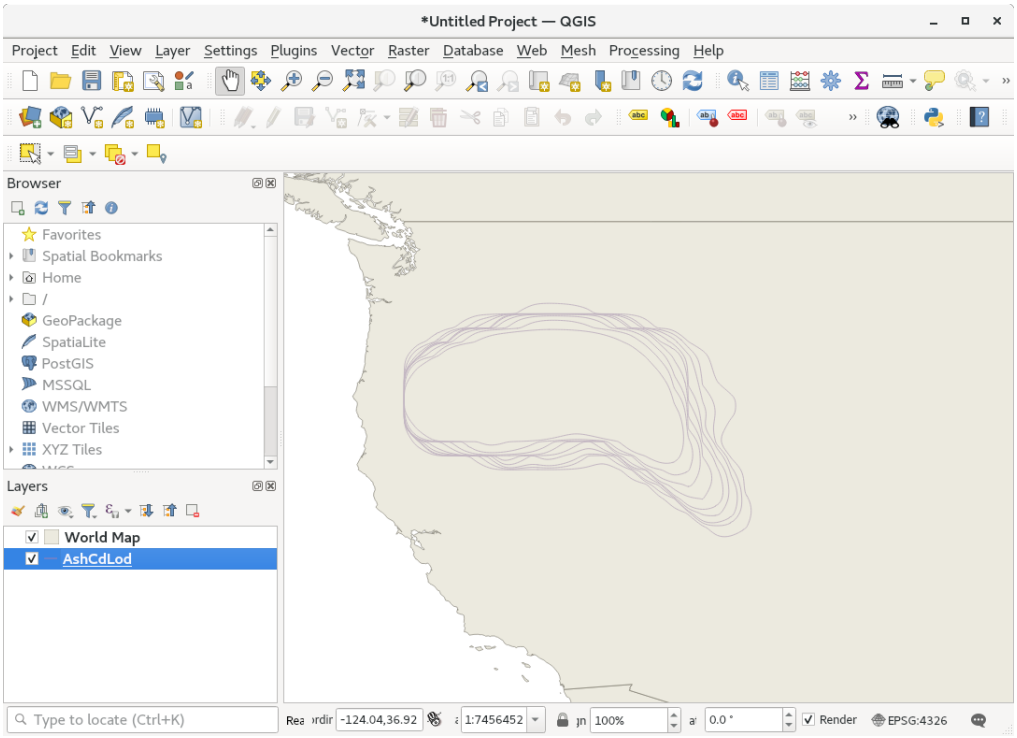
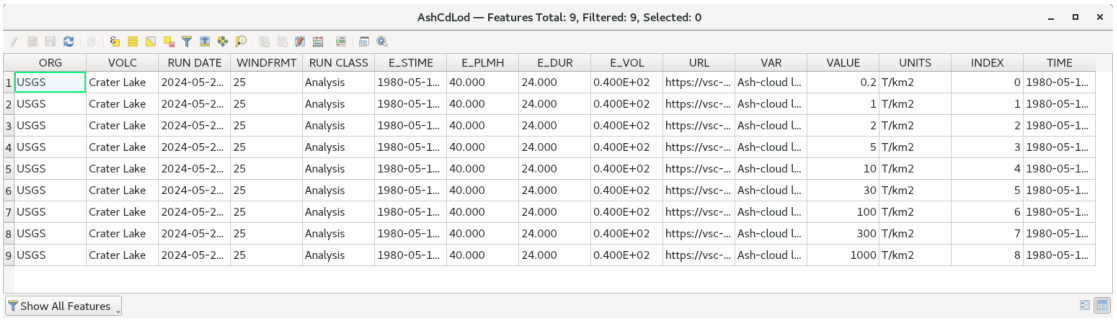


Figure 3.11: Shapefile of cloud load added as a vector layer in QGIS.



	ORG	VOLC	RUN DATE	WINDFRMT	RUN CLASS	E_STIME	E_PLMH	E_DUR	E_VOL	URL	VAR	VALUE	UNITS	INDEX	TIME
1	USGS	Crater Lake	2024-05-2...	25	Analysis	1980-05-1...	40.000	24.000	0.400E+02	https://vsc-...	Ash-cloud l...	0.2	T/km2	0	1980-05-1...
2	USGS	Crater Lake	2024-05-2...	25	Analysis	1980-05-1...	40.000	24.000	0.400E+02	https://vsc-...	Ash-cloud l...	1	T/km2	1	1980-05-1...
3	USGS	Crater Lake	2024-05-2...	25	Analysis	1980-05-1...	40.000	24.000	0.400E+02	https://vsc-...	Ash-cloud l...	2	T/km2	2	1980-05-1...
4	USGS	Crater Lake	2024-05-2...	25	Analysis	1980-05-1...	40.000	24.000	0.400E+02	https://vsc-...	Ash-cloud l...	5	T/km2	3	1980-05-1...
5	USGS	Crater Lake	2024-05-2...	25	Analysis	1980-05-1...	40.000	24.000	0.400E+02	https://vsc-...	Ash-cloud l...	10	T/km2	4	1980-05-1...
6	USGS	Crater Lake	2024-05-2...	25	Analysis	1980-05-1...	40.000	24.000	0.400E+02	https://vsc-...	Ash-cloud l...	30	T/km2	5	1980-05-1...
7	USGS	Crater Lake	2024-05-2...	25	Analysis	1980-05-1...	40.000	24.000	0.400E+02	https://vsc-...	Ash-cloud l...	100	T/km2	6	1980-05-1...
8	USGS	Crater Lake	2024-05-2...	25	Analysis	1980-05-1...	40.000	24.000	0.400E+02	https://vsc-...	Ash-cloud l...	300	T/km2	7	1980-05-1...
9	USGS	Crater Lake	2024-05-2...	25	Analysis	1980-05-1...	40.000	24.000	0.400E+02	https://vsc-...	Ash-cloud l...	1000	T/km2	8	1980-05-1...

Figure 3.12: Shapefile of cloud load shapefile.

In the attribute table shown in Figure 3.12, the fields for ORG, RUN CLASS and URL are the default values, but these can be changed in the optional `RESETPARAMS` block of the control file.

```
**********************
# Reset parameters
**********************
OPTMOD=RESETPARAMS
 cdf_institution      = Volcano Observatory
 cdf_run_class        = Forecast
 cdf_url              = https://volcano-observatory/model-forecasts
********************************************************************************
```

`Ash3d_PostProc` is a convenient tool for plotting the vertical profiles. To plot profiles using the gnuplot graphics package, run:

```
ASH3DPLOT=3 ./Ash3d_PostProc 3d_tephra_fall.nc 16 3
```

which in this case will generate two files for the two requested profile locations: `gnupl_0001.png` and `gnupl_0002.png`. Profile 2 is shown in Figur **??**.

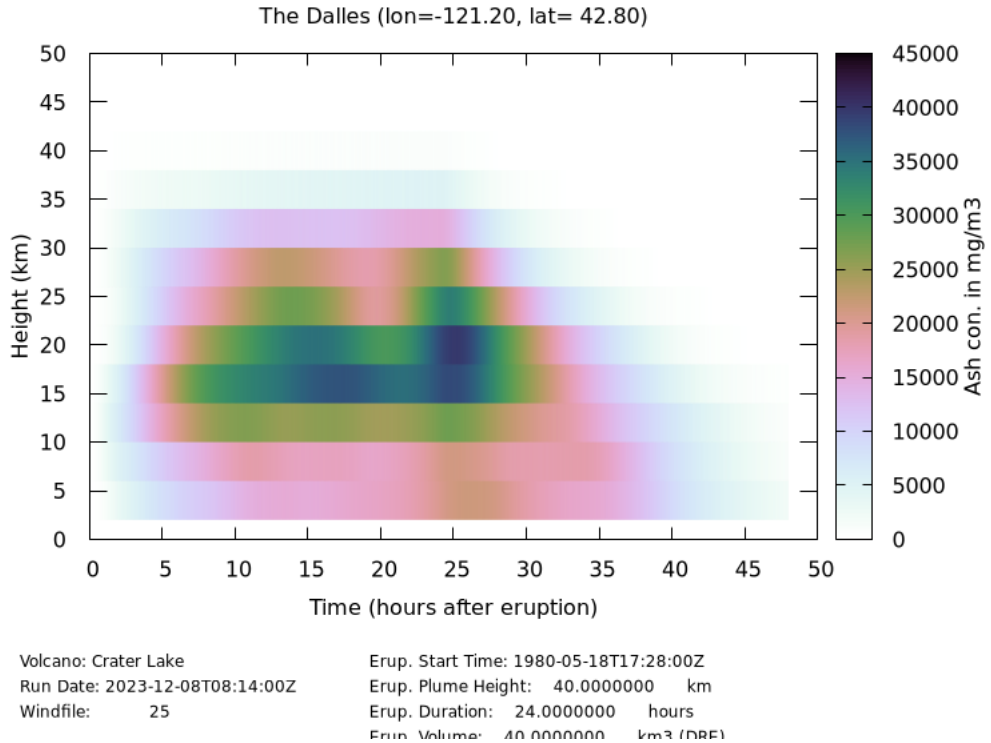


Figure 3.13: Vertical profile of ash concentration at location 2.

Using `Ash3d_PostProc` with command-line arguments is meant to be a quick tool for generating plots from the NetCDF output file. Using a control file, however, allows much more flexibility and allows plotting maps from 2-D ASCII or binary files, or to customize the contour levels and colors.

Below is an example of a control file that can be used to convert a 2-D ESRI ASCII output file of the deposit thickness to a map image with custom contour levels.

```
DepositFile_____final.dat       data_filename
1                       input format code (1=ASCII, 2=binary, 3=NetCDF)
test.inp                Ash3d_control_file_name   (skipped if datafile is NetCDF)
5                       invar [outvar]    input variable and output variable, if different
2 1                     ndims Only needed for format 1 or 2, LatLonFlag
38 20                   nx ny [nz] Also only needed for format 1 or 2
1.0 1.0                 dx dy [dz] Needed if not a part of the data file
225.0 33.0              srtx srty         Start x and y
3                       output format     (1=ASCII, 2=KML 3=image, 4=binary, 5=shapefile)
4                       plot_pref         (1=dislin, 2=plplot, 3=gnuplot, 4=GMT)
-1                      time_step         Only needed if input file is multi-timestep (eg NetCDF)
0                       Filled contour flag (1 for filled, 0 for lines)
1 5                     custom contour flag (1 for true, 0 for false), number of contours
1.0 3.0 10.0 50.0 100.0 lev(ncont)  : contour levels
100 100 100 100 255     R(ncont)    : Red channel of RGB
100 150 200 150   0     G(ncont)    : Green channel of RGB
200 150 100  50   0     B(ncont)    : Blue channel of RGB
```

Line 1 contains the data file name, in this case `DepositFile_____final.dat`, but it could be the output NetCDF file or any 2/3-D binary or ASCII output file. Line 2 specifies the format code for ASCII, binary or NetCDF. Line 3 is the name of the Ash3d control file for this run. This is needed to know the geometry of the simulation, the time steps, volcano name, eruption source parameters, and other information needed for annotating the maps or other output products. If the input data file is the Ash3d NetCDF output file, then this Ash3d input control file is ignored as all the information required is stored in the NetCDF file. Line 4 is the variable code for the input file, which is assumed to be the output variable code unless a second code for the output variable is provided. For example, if the data file is the Ash3d NetCDF data file, then entering '5' in line 4 will result in a plot of the final deposit thickness. If the input data file is a 3-D binary file of the airborne ash concentration, but cloud load is needed for the output, line 4 would contain `1 12`. Line 5 contains the number of dimensions of the data file (2 or 3) and a code indicating if the data are projected or in lon/lat coordinates (0 or 1). `Ash3d_PostProc` currently only can plot projected data using GMT. Line 6 gives the number of nodes in the x,y, and possibly z directions for the data file. Line 7 gives the grid spacing for all the coordinate directions of the file and line 8 gives the start coordinates. Line 8 is the output format code where 1=ASCII, 2=KML 3=image, 4=binary, 5=shapefile. Line 9 is the plotting package preference code: 1=DISLIN, 2=PLplot, 3=gnuplot, 4=GMT. Line 10 is the time step to plot. -1 can be entered to plot the last time step of the file. Line 11 is a flag to indicate if the plot should have contour lines (0) or filled contours (1). This option is not yet fully functional and only contour lines are available. Line 12 starts with a flag allowing custom contour levels (1) to override the default levels for that variable. If this flag is set, a second value is read, giving the number of custom levels (`nlev`). If custom contours are requested, then four additional lines are read in the control file. Line 13 contains the `nlev` floating point values for the contour levels in the default units of the

output variable. Lines 14, 15, and 16 are the `nlev` integer values of the RGB components of the colors (0-255).

3.3.2 Generating maps with GMT

There are many programs that can be used for generating maps from Ash3d output. We primarily use Generic Mapping Tools (GMT) due to its flexibility, customization and ease of use in bash scripts. Plots generated on the Ash3d web-servers use scripts that read information about the Ash3d run from the NetCDF file, dynamically plot data, select the top most populous cities in the domain for annotating, and evaluate legend placement based on deposit location. These scripts are described in more detail in Chapter 5.

There are many tutorials online for learning GMT. Here we just show an example script that can be used to generate a basemap, display shaded topography and overlay deposit contours. The resulting plot is shown in 3.14.

```
#!/bin/bash

AREA=-R-80.0/-70.0/2.0/8.0
PROJ=-JM-75.0/4.0/20
BASE=-Ba2g2/2g1

gmt makecpt -Cetopo1 -D -V -T-3500/3500/10 > topo.cpt

gmt grdconvert gebco_2023_n20.0_s-10.0_w-100.0_e-50.0.nc?elevation fulltopo.grd
gmt grdcut fulltopo.grd -Gtopo.grd $AREA
gmt grdgradient topo.grd -Gtopo.grad -A90/20 -fg

gmt pscoast $BASE $AREA $PROJ -Dh -G -K > temp.ps
gmt grdimage topo.grd -Itopo.grad $AREA $PROJ -Ctopo.cpt -K -O >> temp.ps
gmt pscoast -Q $BASE $AREA $PROJ -Dh -K -O >> temp.ps

echo "0.01   C" > dpm_0.01.lev   #deposit (0.01 mm)
echo "0.03   C" > dpm_0.03.lev   #deposit (0.03 mm)
echo "0.1    C"  > dpm_0.1.lev   #deposit (0.1 mm)
echo "0.3    C"  > dpm_0.3.lev   #deposit (0.3 mm)
echo "1.0    C"  >   dpm_1.lev   #deposit (1 mm)
echo "3.0    C"  >   dpm_3.lev   #deposit (3 mm)
echo "10.0   C"  >  dpm_10.lev   #deposit (1 cm)
echo "30.0   C"  >  dpm_30.lev   #deposit (3 cm)
echo "100.0  C"  > dpm_100.lev   #deposit (10cm)
echo "300.0  C"  > dpm_300.lev   #deposit (30cm)

infile="../3d_tephra_fall_ERA5_Topo.nc"
##get volcano longitude, latitude
VCLON=`ncdump -h ${infile} | grep b1l5 | cut -d\" -f2 | awk '{print $1}'`
VCLAT=`ncdump -h ${infile} | grep b1l5 | cut -d\" -f2 | awk '{print $2}'`
dep_grd="dep_fin.grd"
gmt grdconvert "$infile?depothickFin" ${dep_grd}
gmt grdcontour ${dep_grd}   $AREA $PROJ $BASE -Cdpm_0.01.lev -A- -W3,214/222/105   -O -K >> temp.ps
gmt grdcontour ${dep_grd}   $AREA $PROJ $BASE -Cdpm_0.03.lev -A- -W3,249/167/113   -O -K >> temp.ps
gmt grdcontour ${dep_grd}   $AREA $PROJ $BASE -Cdpm_0.1.lev  -A- -W3,128/0/128   -O -K >> temp.ps
```

```
gmt grdcontour ${dep_grd}   $AREA $PROJ $BASE -Cdpm_0.3.lev  -A- -W3,0/0/255     -O -K >> temp.ps
gmt grdcontour ${dep_grd}   $AREA $PROJ $BASE -Cdpm_1.lev    -A- -W3,0/128/255   -O -K >> temp.ps
gmt grdcontour ${dep_grd}   $AREA $PROJ $BASE -Cdpm_3.lev    -A- -W3,0/255/128   -O -K >> temp.ps
gmt grdcontour ${dep_grd}   $AREA $PROJ $BASE -Cdpm_10.lev   -A- -W3,195/195/0   -O -K >> temp.ps
gmt grdcontour ${dep_grd}   $AREA $PROJ $BASE -Cdpm_30.lev   -A- -W3,255/128/0   -O -K >> temp.ps
gmt grdcontour ${dep_grd}   $AREA $PROJ $BASE -Cdpm_100.lev  -A- -W3,255/0/0     -O -K >> temp.ps
gmt grdcontour ${dep_grd}   $AREA $PROJ $BASE -Cdpm_300.lev  -A- -W3,128/0/0     -O -K >> temp.ps

# Last gmt command is to plot the volcano and close out the ps file
echo $VCLON $VCLAT '1.0' | gmt psxy $AREA $PROJ -St0.1i -Gblack -Wthinnest -O >> temp.ps
```

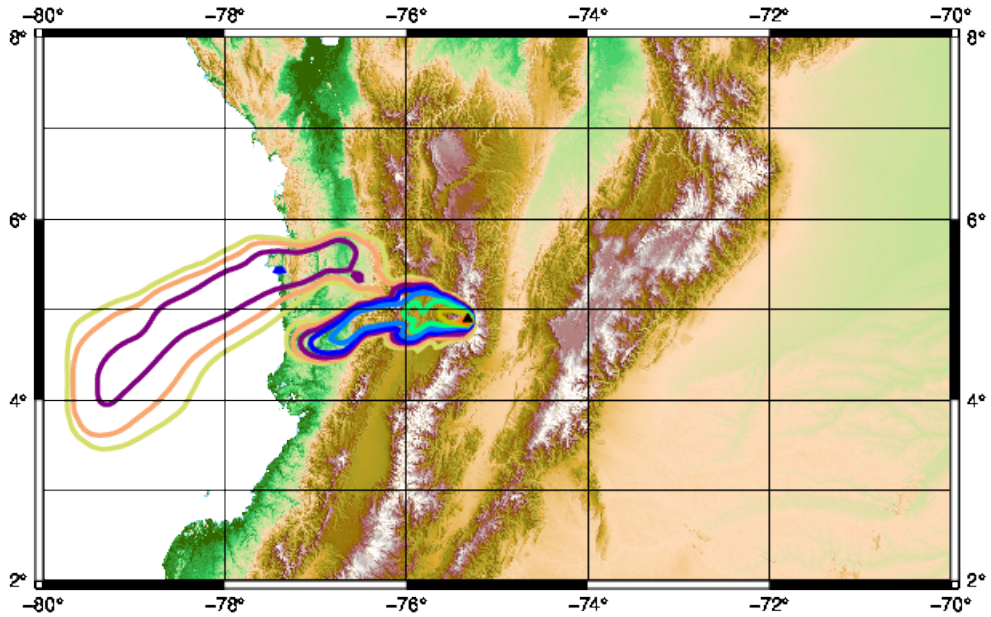


Figure 3.14: GMT map of Ash3d deposit for Nevado del Ruiz

Chapter 4

Software structure and optional modules

Ash3d has been under development since 2009 and has evolved to suit the needs of both operational forecasting as well as a research tool. As research ideas were tested and bits of code were altered and inserted at various points in the software, this evolution lead to dormant branches of the code, inefficiencies, changes in default values that became part of the operational branch, and occasionally the introduction of bugs in the code. To mitigate these problems and to facilitate the development of new features or research ideas without affecting the core functionality of the software, we began an effort to re-structure the software. The restructuring has been guided by several goals: to group and isolate common-themed functions and variables into modules, to separate reusable code into libraries, to optimize structure of and access to the main variables, to isolate essential functionality into a core set of subroutines that would require minimal continued development, and to move all the essential control structure of the program to the top-level file (`Ash3d.F90`). This software structure would be a minimal 'core code' to be available for operational usage or for research efforts requiring only essential functionality. If a non-essential feature were needed to address a question (e.g. predicting the resuspension of deposited ash, tracking the advection of fallout with ocean surface currents), then this feature could be isolated in an optional module which could be compiled with a version of the `Ash3d.F90` linking to all the core-code files.

The structure of the core code is outlined in Figure 4.1. The files are broadly sorted into those needed for setup/initialization, calculation of meteorological values, source terms, advection routines, diffusion routines, and output. Files that define modules are encircled by bold lines. Files that need direct access to the main array of concentration values (`concen_pd`) are colored orange. The number of files with access to `concen_pd` is minimized since repeated access to this array can effect performance, particularly on hybrid GPU/CPU systems.

The basic flow of the code is as follows:

1. Initialize Run
   - Test runtime environment (variables, OpenMP, test system configuration)
   - Parse command-line / read control file

Figure 4.1: Ash3d program structure.

- Read auxiliary data: airport/POI and volcano lists
- Calculate computational mesh parameters
- Allocate memory
- Initialize concentration array (possibly with source)
- Read meteorological files for starting values
- Initialize output files

2. Solve advection-diffusion equations
   - Start time loop
   - Update atmospheric data for current time
   - Calculate source term
   - Set boundary conditions
   - Advect horizontally
   - Advect vertically
   - Diffuse vertically
   - Diffuse horizontally
   - Test for output criteria (log-step, output step)
   - Write output data if needed
   - Check stop conditions (end time reached, no ash aloft, etc.)
   - Integrate solution forward in time
3. Closing routines
   - Finalize output
   - Deallocate all memory

4.1 User-customization

Although Ash3d is not a particularly large program, understanding all the details of the software is a cumbersome and largely unnecessary task. In order to modify Ash3d to include a new feature (source term, evolution or removal mechanism, etc.), the recommended approach is to create the new routines in a separate module and edit a copy of the main program file `Ash3d.F90` to include calls to these routines. The makefile can be edited to build this new module and link with the objects in the core-code repository. `Ash3d.F90` has commented sections throughout that indicate where edits would be needed to link to the new module.

An example of using an optional module is given in the git repository https://github.com/usgs/hschwaiger-usgs/Ash3d_OptionalModules. The makefile in this repository contains the variable `ASH3dCCSRC=/work/USGS/Software_repos/GIT/Ash3d/src` which points to the location of the Ash3d source files. The optional module included in this repository is for

running the various test cases to test the convergence properties of the code (see Section F.3). The main program file is replaced with `Ash3d_TC.F90` which contain all the function calls needed to set up and run the test cases. In this example, the additional code is invoked using pre-processor directives, though this is not necessary. This module is not especially invasive and only requires a few subroutine calls: a call to `set_TestCase_globvars` which sets global variables turning on/off advection and diffusion routines, a call to `set_TestCase_windfield` and `DistSource` to specify the wind field and initialize the source, a disabling of `MassFluxCalculator`, and finally a call to `Testcase_CalcErrors` which prepares the output data. The goal is to have all additional subroutine callable from the top level file (`Ash3d_TC.F90`) with all others linked to the core repository of the Ash3d code. In this example, `Set_BC.f90` needed to be modified for the test case using the Method of Manufactured Solution as is replaced with `Set_BC_TC.f90`. Details of this module along with instructions on running the scripts can be found in Appendix F.

For a more general case, an example is given for the module for the accelerated removal mechanisms of ash due to hydrometeors (e.g. wet deposition). As before, the only files that need to be modified from the core-code Ash3d version is the top-level file (a copy of `Ash3d.F90` and the `makefile`. Additionally, the module containing all the public variables and routines is needed. There are several subroutines that are generally needed. Firstly, if a dedicated block of the input file is to be read for module-specific input values, a subroutine such as `input_data_WetDepo` should be included.

```
        ! input data for ash transport
      call Read_Control_File
!-------------------------------------------------------------------------------
!       OPTIONAL MODULES
!         Insert calls to custom input blocks here

      write(global_info,*)&
        "Now looping through optional modules found in input file"
      do j=1,nmods
        write(global_info,*)"Testing for ",OPTMOD_names(j),j
#ifdef WETDEPO
        if(OPTMOD_names(j).eq.'WETDEPO')then
          write(global_info,*)"  Reading input block for WETDEPO"
          call input_data_WetDepo
        endif
#endif
!
!-------------------------------------------------------------------------------
```

The structure of this optional module block can be designed to the user's needs and can have as many lines as necessary to fully characterize all the necessary parameters. In this case, for example, we might want have fields to enable or disable different removal mechanisms via hydrometeors (rain scavenging, cloud condensation nucleation, etc.). These mechanisms might have parameters associated with scavenging rate or a different fall mechanics. Maybe there are specialize output variables that might optionally be included in the output file that could be toggled on or off via settings in this block.

Next, the space needed for storing module-specific variables (some on the computational grid and some only on the meteorological grid) needs to be allocated.

```
!-------------------------------------------------------------------------------
!       OPTIONAL MODULES
!         Insert calls to optional variable allocation subroutines here
!
#ifdef WETDEPO
      if(USE_WETDEPO)then
        call Allocate_WetDepo_global
        call Allocate_WetDepo_Met
      endif
#endif
```

In principle, the variables on the meteorological grid and on the computational grid could be allocated in the same subroutine, but we generally separate them for clarity.

If any of the calculations rely on fields read from the numerical weather prediction files, then we need a subroutine to read and interpolate these data. The list of variables available to read is given in Table 1 of Appendix 8.1 of the MetReader documentation.

```
!-------------------------------------------------------------------------------
!       OPTIONAL MODULES
!         Insert calls to special MesoInterpolaters subroutines here
!
#ifdef WETDEPO
        if(USE_WETDEPO)     call Set_WetDepo_Meso(Load_MesoSteps,Interval_Frac)
#endif
```

For example, the scavenging of particles from the atmosphere is dependent on the precipitation rate (variable code `44` from the MetReader documentation). So to calculate the scavenging coefficient at each time-step, one would read the precipitation rate for the bracketing NWP time steps, interpolate to the computational grid, then calculate the scavenging coefficients on the computational grid at the meteorological time steps, then finally interpolating to the current time:

```
ivar = 44 !  Precipitation rate large-scale (liquid)
call MR_Read_2d_Met_Variable(ivar,MR_iMetStep_Now)
call MR_Regrid_MetP_to_CompGrid(MR_iMetStep_Now)
precipitation_rate_2d(:,:)   = MR_dum2d_comp(:,:)
call Set_Scav_Coeff(MR_iMetStep_Now,1)

call MR_Read_2d_Met_Variable(ivar,MR_iMetStep_Now+1)
call MR_Regrid_MetP_to_CompGrid(MR_iMetStep_Now+1)
precipitation_rate_2d(:,:)   = MR_dum2d_comp(:,:)
call Set_Scav_Coeff(MR_iMetStep_Now+1,2)

call Interpolate_WetDepo(Interval_Frac)
```

If any variables are needed for output to the NetCDF file, the list of output variables in the module `Output_Vars` can be appended. These variables in `Output_Vars` are sorted by their dimensionality:

```
var_User2d_static_XY  ! 2d variable (nx,ny)
var_User2d_XY         ! 3d variably (nx,ny,nt)
var_User3d_XYGs       ! 4d variable (nx,ny,ns,nt)
var_User3d_XYZ        ! 4d variable (nx,ny,nz,nt)
var_User4d_XYZGs      ! 5d variable (nx,ny,nz,ns,nt
```

A subroutine for preparing the master list of variables, their names, units, FillValues, etc. should be called prior to the initialization of the output file. For example, if only the 2-D surface precipitation and the 4-D scavenging coefficients are added to the output file, the module should include:

```
nvar_User2d_XY_WetDepo    = 1
nvar_User4d_XYZGs_WetDepo = 1
ivar = 1
temp_2d_name_WetDepo(ivar)  = "p0"
temp_2d_lname_WetDepo(ivar) = "Precipitation rate @ surf"
temp_2d_unit_WetDepo(ivar)  = "kg/m2/s"
temp_2d_MissVal_WetDepo(ivar) = -9999.0_op
temp_2d_FillVal_WetDepo(ivar) = -9999.0_op

ivar = 1
temp_4d_name_WetDepo(ivar)  = "ScavCo_liq"
temp_4d_lname_WetDepo(ivar) = "Scavenging coefficient"
temp_4d_unit_WetDepo(ivar)  = "1/hr"
temp_4d_MissVal_WetDepo(ivar) = -9999.0_op
temp_4d_FillVal_WetDepo(ivar) = -9999.0_op
nvar_User2d_XY    = nvar_User2d_XY        + nvar_User2d_XY_WetDepo
nvar_User4d_XYZGs = nvar_User4d_XYZGs     + nvar_User4d_XYZGs_WetDepo
```

Then to actually update the master list of output variables in `Output_Vars.f90`, use a subroutine such as:

```
!------------------------------------------------------------------------
!       OPTIONAL MODULES
!         Insert calls to prep user-specified output
!
#ifdef WETDEPO
      if(USE_WETDEPO) call Prep_output_WetDepo
#endif
```

with code to update the master lists:

```
do i=1,nvar_User4d_XYZGs_WetDepo
  indx = indx_User4d_XYZGs_WetDepo+i
  var_User4d_XYZGs_name(indx)   = temp_4d_name_WetDepo(i)
  var_User4d_XYZGs_unit(indx)   = temp_4d_unit_WetDepo(i)
  var_User4d_XYZGs_lname(indx)  = temp_4d_lname_WetDepo(i)
  var_User4d_XYZGs_MissVal(indx)= temp_4d_MissVal_WetDepo(i)
  var_User4d_XYZGs_FillVal(indx)= temp_4d_FillVal_WetDepo(i)
  if(i.eq.1) &
```

```
  var_User4d_XYZGs(1:nxmax,1:nymax,1:nzmax,1:n_gs_max,indx) = &
  real(scav_coeff_Liq_3d(1:nxmax,1:nymax,1:nzmax,1:n_gs_max),kind=op)
enddo
```

After the start of the time loop, the interpolation routine that calculates variables on the computational grid at the current time step from the NWP files need to be called.

```
          ! find the wind field at the current time
        first_time     = .false.
        call MesoInterpolater(time , Load_MesoSteps , Interval_Frac, first_time)
!-------------------------------------------------------------------------------
!       OPTIONAL MODULES
!         Insert calls to special MesoInterpolaters subroutines here
!
#ifdef WETDEPO
        if(USE_WETDEPO)     call Set_WetDepo_Meso(Load_MesoSteps,Interval_Frac)
#endif
!-------------------------------------------------------------------------------
```

Within the time loop, any physics that effects the simulation can be called. For the wet removal of airborne tephra, for example, a concentration-dependent decay of the ash is calculated through sub-stepping (due to the stiffness of the decay equations) over the full time step.

```
!-------------------------------------------------------------------------------
!       OPTIONAL MODULES
!         Insert calls to optional deposition routines here
!
#ifdef WETDEPO
        if(USE_WETDEPO)then
          call Wet_Depo_Rainout
        endif
#endif
!-------------------------------------------------------------------------------
```

Then, any output variables can be calculated with the corresponding `var_User2d_[---]` variables updated. These will be written to the NetCDF file, if that output format is specified in the Ash3d control file.

```
!-------------------------------------------------------------------------------
!       OPTIONAL MODULES
!         Insert calls output routines (every log-step) here
!
#ifdef WETDEPO
              if(USE_WETDEPO) call ThicknessCalculator_WetDepo
#endif
!-------------------------------------------------------------------------------
```

And finally after the completion of the time-step loop, routines for freeing the allocated memory should be called.

```
      ! clean up memory
      call dealloc_arrays
!-----------------------------------------------------------------------------
!       OPTIONAL MODULES
!         Insert calls deallocation routines here
!
#ifdef WETDEPO
      if(USE_WETDEPO)then
        call Deallocate_WetDepo_global
        call Deallocate_WetDepo_Met
      endif
#endif
!-----------------------------------------------------------------------------
```

Except in unusual circumstances, these subroutine should be called from the replaced top-level file, (e.g. `Ash3d_WetDepo.F90`).

Chapter 5

Additional tools/Examples

Appendix A

Example Auxiliary Files

A.1 Input Files

```
*******************************************************************************
1992 08 19   1.0   3.5     13.7   0.014
*******************************************************************************
```

Appendix B

Atmospheric data file formats

B.1 1-D ASCII profiles

B.2 NetCDF template reader

Appendix C

Finite Volume Solvers

C.1 Grid Geometry

Volume:

$$\kappa_{i,j,k} = \int_{r_K}^{r_{K+1}} \int_{\theta_J}^{\theta_{J+1}} \int_{\lambda_I}^{\lambda_{I+1}} r \sin\theta \mathrm{d}\lambda \, r \mathrm{d}\theta \, \mathrm{d}r = (\lambda_{I+1} - \lambda_I) \frac{1}{3} \left(r_{K+1}^3 - r_K^3\right) (\cos\theta_J - \cos\theta_{J+1}) \tag{C.1}$$

Surface area:

$$\sigma_x = \sigma_{I,j,k} = \int_{r_K}^{r_{K+1}} \int_{\theta_J}^{\theta_{J+1}} r \mathrm{d}\theta \, \mathrm{d}r = (\theta_{J+1} - \theta_J) \frac{1}{2} \left(r_{K+1}^2 - r_K^2\right) \tag{C.2}$$

$$\sigma_y = \sigma_{i,J,k} = \int_{r_K}^{r_{K+1}} \int_{\lambda_I}^{\lambda_{I+1}} r \sin\theta \mathrm{d}\lambda \, \mathrm{d}r = \sin\theta \, (\lambda_{I+1} - \lambda_I) \frac{1}{2} \left(r_{K+1}^2 - r_K^2\right) \tag{C.3}$$

$$\sigma_z = \sigma_{i,j,K} = \int_{\lambda_I}^{\lambda_{I+1}} \int_{\theta_J}^{\theta_{J+1}} r \sin\theta \mathrm{d}\lambda \, r \mathrm{d}\theta = r_K^2 \, (\lambda_{I+1} - \lambda_I) \, (\cos\theta_J - \cos\theta_{J+1}) \tag{C.4}$$

C.2 Advection Routines

Ash3d has been written to accommodate different advection routines such as Donor-Cell-Update (DCU), Corner Transport Update (CTU), and Semi-Lagrange (SL). In this public version of the software, only DCU is included.

The implementation of DCU follows closely with the equations presented in [10].

Start with the fluctuation form (Eq. 6.59 from [10]) of the update in Q to a cell i given below.

$$Q_i^{n+1} = Q_i^n - \frac{\Delta t}{\Delta x} \left(\mathcal{A}^- \Delta Q_{i+\frac{1}{2}} + \mathcal{A}^+ Q_{i-\frac{1}{2}}\right) - \frac{\Delta t}{\Delta x} \left(\widetilde{F}_{i+\frac{1}{2}} - \widetilde{F}_{i-\frac{1}{2}}\right) \tag{C.5}$$

Note that this is analogous to the flux form with $(A^-\Delta Q_{i+\frac{1}{2}} + A^+\Delta Q_{i-\frac{1}{2}})$ with the left and right fluctuations at an interface satisfying Eq. 6.57

$$(\mathcal{A}^-\Delta Q_{i-\frac{1}{2}} + \mathcal{A}^+\Delta Q_{i-\frac{1}{2}}) = f(Q_i) - f(Q_{i-1}) = A\Delta Q_{i-\frac{1}{2}} \tag{C.6}$$

We define $\mathcal{A}^-$ and $\mathcal{A}^+$ according to Eq. 9.38 for the conservative variable coefficient case.

$$\mathcal{A}^+\Delta Q_{i-\frac{1}{2}} = F_i - F_{i-\frac{1}{2}} \tag{C.7}$$

$$\mathcal{A}^-\Delta Q_{i-\frac{1}{2}} = F_{i-\frac{1}{2}} - F_{i-1} \tag{C.8}$$

where $F_{i-\frac{1}{2}}$ are the interface fluxes and F_i are the cell-centered representations. These cell-centered values are somewhat arbitrary since they cancel each other in Eq. C.5, but we use $F_i = (u^+_{i-\frac{1}{2}} + u^-_{i+\frac{1}{2}})Q_i$ as suggested by Eq 9.39 of [10]. Interface fluxes are given by (the un-numbered equation of [10] between 9.35 and 9.36)

$$F_{i-\frac{1}{2}} = u^+_{i-\frac{1}{2}}Q_{i-1} + u^-_{i-\frac{1}{2}}Q_i \tag{C.9}$$

So

$$\mathcal{A}^+Q_{i-\frac{1}{2}} = \left(u^+_{i-\frac{1}{2}}Q_i + u^-_{i+\frac{1}{2}}Q_i\right) - \left(u^+_{i-\frac{1}{2}}Q_{i-1} + u^-_{i-\frac{1}{2}}Q_i\right) \tag{C.10}$$

$$= u^+_{i-\frac{1}{2}}(Q_i - Q_{i-1}) + Q_i\left(u^-_{i+\frac{1}{2}} - u^-_{i-\frac{1}{2}}\right) \tag{C.11}$$

$$\mathcal{A}^-Q_{i-\frac{1}{2}} = \left(u^+_{i-\frac{1}{2}}Q_{i-1} + u^-_{i-\frac{1}{2}}Q_i\right) - \left(u^+_{i-\frac{3}{2}}Q_{i-1} - u^-_{i-\frac{1}{2}}Q_{i-1}\right) \tag{C.12}$$

$$= u^-_{i-\frac{1}{2}}(Q_i - Q_{i-1}) + Q_{i-1}\left(u^+_{i-\frac{1}{2}} - u^+_{i-\frac{3}{2}}\right) \tag{C.13}$$

The higher order terms of Eq C.5 above are given by Eq. 6.60 of [10].

$$\widetilde{F}_{i-\frac{1}{2}} = \frac{1}{2}\sum_{p=1}^{m}\left|s^p_{i-\frac{1}{2}}\right|\left(1 - \frac{\Delta t}{\Delta x}\left|s^p_{i-\frac{1}{2}}\right|\right)\widetilde{W}^p_{i-\frac{1}{2}} \tag{C.14}$$

where $\widetilde{W}^p_{i-\frac{1}{2}}$ are the limited p-waves from an interface and $s^p_{i-\frac{1}{2}}$ are their speeds. In our case, we have $p = 1$, $s^p_{i-\frac{1}{2}} = u_{i-\frac{1}{2}}$ and $\widetilde{W}_{i-\frac{1}{2}}$ is the limited version of the wave, .i.e $\widetilde{\Delta Q}$. Eq. C.14 reduces to

$$\widetilde{F}_{i-\frac{1}{2}} = \frac{1}{2}\left|u_{i-\frac{1}{2}}\right|\left(1 - \frac{\Delta t}{\Delta x}\left|u_{i-\frac{1}{2}}\right|\right)\widetilde{\Delta Q}_{i-\frac{1}{2}} \tag{C.15}$$

The limited wave, $\widetilde{\Delta Q}$, can be set by first determining the function

$$\theta = \frac{\Delta Q_{\text{upwind}}}{\Delta Q_{i-\frac{1}{2}}} \tag{C.16}$$

then setting $\widetilde{\Delta Q} = \phi\left(\theta_{i-\frac{1}{2}}\right)\Delta Q_{i-\frac{1}{2}}$ (Eq. 6.34 of [10]):

- No limiter: $\phi(\theta) = 0 \Rightarrow \widetilde{\Delta Q}_{i-\frac{1}{2}} = 0$ (Note: this sets $\widetilde{F}_{i-\frac{1}{2}} = 0$)
- Lax-Wendrof: $\phi(\theta) = 1 \Rightarrow \widetilde{\Delta Q}_{i-\frac{1}{2}} = (Q_i - Q_{i-1})$
- Beam-Warming: $\phi(\theta) = \theta \Rightarrow \widetilde{\Delta Q}_{i-\frac{1}{2}} = \Delta Q_{\text{upwind}}$
- Fromm: $\phi(\theta) = \frac{1}{2}(1+\theta)$
- MinMod: $\phi(\theta) = \text{minmod}(1, \theta)$
- SuperBee: $\phi(\theta) = \max(0, \min(1, 2\theta), \min(2, \theta))$
- MC: $\phi(\theta) = \max\left(0, \min\left(\frac{(1+\theta)}{2}, 2, 2\theta\right)\right)$

To allow curvilinear coordinates, we replace the Δx in the above equations with κ_i and scale velocities by the interface area $\sigma_{i-\frac{1}{2}}$, since these velocities are only needed at the interfaces.

C.2.1 Numerical implementation

Variables are generally associated with cell-centers and are indexed by i. These include Q, u, and κ. In the equations above, several variables are needed on interfaces. We use the following indexing for interface $I = i - \frac{1}{2}$ and define

$$\begin{aligned} \sigma_I &= \text{area of interface} \\ \bar{u}_I &= u_{i-\frac{1}{2}} & &= \frac{1}{2}(u_i + u_{i-1}) \\ \Delta Q_I &= \Delta Q_{i-\frac{1}{2}} & &= Q_i - Q_{i-1} \end{aligned}$$

On the first loop over I, the following are calculated:

$$\begin{aligned} \texttt{ubar\_I(1:ncells+1)} &= \bar{u}_I \\ \texttt{usig\_I(1:ncells+1)} &= (\bar{u}\sigma)_I \\ \texttt{dqi\_I(1:ncells+1)} &= \Delta Q_I \\ \texttt{ldq} &= \widetilde{\Delta Q}_I \\ \texttt{fss\_I(1:ncells+1)} &= \widetilde{F}_I \end{aligned}$$

where

$$\widetilde{F}_I = \frac{1}{2}|\bar{u}_I|\left(1 - |(\bar{u}\sigma)_I|\frac{\Delta t}{\kappa_i}\right)\widetilde{\Delta Q}_I \tag{C.17}$$

On the second pass over I, $\mathcal{A}^{\pm}\Delta Q_I$ are calculated according to

$$\begin{aligned} \texttt{fs\_I(1:ncells+1,1)} = \mathcal{A}^{-}Q_I &= \bar{u}_I^{-}\Delta Q_I + Q_{i-1}(\bar{u}_I^{+} - \bar{u}_{I-1}^{+}) \\ \texttt{fs\_I(1:ncells+1,2)} = \mathcal{A}^{+}Q_I &= \bar{u}_I^{+}\Delta Q_I + Q_i(\bar{u}_{I+1}^{-} - \bar{u}_I^{-}) \end{aligned}$$

Note that the first term of these equations uses the interface concentration jump and left/right interface concentration while the second terms using cell-centered concentrations multiplied by the differences in interface velocities across that cell-centered value. Therefore, calculating `fs_I` at interface I requires cell-centered velocities at $i-2, i-1, i, i+1, i+2$. Similarly, some limiters also require concentrations on these cells.

Lastly, we loop of the cell index i and build the four components of the update:

- Rightward fluctuation at left boundary = `RFluct_Lbound` = $\mathcal{A}^+ \Delta Q_I$
- Leftward fluctuation at right boundary = `LFluct_Rbound` = $\mathcal{A}^- \Delta Q_{I+1}$
- Limited-q at left boundary = `LimFlux_Lbound` = $\widetilde{F}_I$
- Limited-q at right boundary = `LimFlux_Rbound` = $\widetilde{F}_{I+1}$

C.3 Diffusion Routines

Ash3d uses two implementations of the diffusion term: the explicit finite difference discretization and the implicit Crank-Nicolson method.

C.3.1 Explicit finite difference

Using Eq. 4.11 of [10]:

$$Q_i^{n+1} = Q_i^n + \frac{\Delta t}{\Delta x^2}\left[k_{i+\frac{1}{2}}\left(Q_{i+1}^n - Q_i^n\right) - k_{i-\frac{1}{2}}\left(Q_i^n - Q_{i-1}^n\right)\right] \tag{C.18}$$

Again, we use the following indexing for interface $I = i - \frac{1}{2}$ and define

$$\begin{aligned} \sigma_I &= \text{area of interface} \\ \bar{k}_I = k_{i-\frac{1}{2}} &= \frac{1}{2}\left(k_i + k_{i-1}\right) \\ \Delta Q_I = \Delta Q_{i-\frac{1}{2}} &= Q_i - Q_{i-1} \end{aligned}$$

Similar to the advection routines, we generalize Eq. C.18 to curvilinear coordinates by replacing Δx with κ and scaling the interface ΔQ_I by the square of the interface area σ_I^2 leading to

$$Q_i^{n+1} = Q_i^n + \frac{\Delta t}{\kappa_i^2}\left[\left(k\sigma^2\Delta Q^n\right)_{I+1} - \left(k\sigma^2\Delta Q^n\right)_I\right] \tag{C.19}$$

Note that this method is stable when $\Delta t = \mathcal{O}(\Delta x^2)$

C.3.2 Implicit Crank-Nicolson

Using Eq. 4.13 of [10]:

$$Q_i^{n+1} = Q_i^n + \frac{\Delta t}{2\Delta x^2}\left[k_{I+1}\Delta Q_{I+1}^n - k_I\Delta Q_I^n + k_{I+1}\Delta Q_{I+1}^{n+1} - k_I\Delta Q_I^{n+1}\right] \tag{C.20}$$

or after modifying for curvilinear coordinates

$$Q_i^{n+1} = Q_i^n + \frac{\Delta t}{2\kappa_i^2}\left[\left(k\sigma^2\Delta Q^n\right)_{I+1} - \left(k\sigma^2\Delta Q^n\right)_I + \left(k\sigma^2\Delta Q^{n+1}\right)_{I+1} - \left(k\sigma^2\Delta Q^{n+1}\right)_I\right] \tag{C.21}$$

C.3.3 Numerical implementation

Appendix D

Fall dynamics

There are several models of the fall dynamics of ash particles that have been implemented in Ash3d. All calculate the fall velocity of a particle according to the following equation [2, p.182]:

$$v_s = \sqrt{\frac{4d\rho g}{3C_d\rho_a}} \tag{D.1}$$

This formula requires a particle diameter and density (d and ρ), air density ρ_a, and the drag coefficient, C_d. The various fall models differ in their specification of C_d, but generally are functions of Re and are variations of the Stokes Law for spherical particle at low Re.

$$C_d = \mathcal{F}(\mathrm{Re}) = \frac{24}{\mathrm{Re}} \tag{D.2}$$

The particle Reynolds number Re, is given by

$$\mathrm{Re} = \frac{v_s\rho_a d}{\eta_a} \tag{D.3}$$

where η_a is the viscosity of air. Given an arbitrary function for C_d, the three equations D.1 through D.3 require an iterative approach to solving for v_s. Ash3d implements this solver by initializing the fall velocity to 1 m/s, solving for the corresponding Re via D.3, then calculating C_d via the particular fall model, and then updating v_s via D.1. This process repeats until v_s changes by less than 0.1%.

Several atmospheric quantities are needed to calculate the fall velocity, primarily the density and viscosity of air. The density is calculated via the ideal gas law and the viscosity via Sutherland's Law [9, p.102, Eq.4.54]

$$\eta_a = 1.8325 \times 10^{-5} \left(\frac{416.16}{T + 120}\right)\left(\frac{T}{296.16}\right)^{1.5} \tag{D.4}$$

For conditions where ρ_a is low and particle size is small, the mean free path of a molecule in the atmosphere, λ_a, can be on the scale of the particle diameter. For these conditions

an additional modification to the expression for C_d can be invoked by scaling C_d by the the Cunningham slip-correction factor

$$C_c = 1 + \mathrm{Kn}\left[\alpha + \beta \exp\left(-\frac{\gamma}{\mathrm{Kn}}\right)\right] \tag{D.5}$$

where the particle Knudsen number is given by $\mathrm{Kn} = 2\lambda_a/d$. Ash3d uses values for the empirical coefficients α, β, and γ given by [20, p.407, Eq.9.34]. The air mean free path, λ_a, is given by [20, p.399, Eq.9.6]:

$$\lambda_a = \frac{2\eta_a}{P\sqrt{8M_B/\pi RT}} \tag{D.6}$$

For spherical particles in the low Re region, the Stokes fall model (`FV_ID=5`) can be used, and C_d is calculated according to Eq. D.2. For non-spherical particles, Ash3d has implemented the Wilson and Huang model (with optional modifications for slip and following Pfeiffer et al) and the Ganser model.

The Wilson and Huang model uses C_d according to

$$C_d = \frac{24}{\mathrm{Re}} F^{-0.828} + 2\sqrt{1.07 - F} \tag{D.7}$$

where F is a shape parameter for an ellipsoidal particle defined as the ratio of the average of the minor axes to the major axis of the particle: $(B + C)/2A$.

If using the Cunningham slip-flow correction, for ellipsoidal particles, Eq. D.5 must be adjusted by scaling Kn according to [**?**, Table 1].

Equation D.7 was empirically determined from fall data from volcanic particles in the size range $30\,\mu\mathrm{m} < d < 1500\,\mu\mathrm{m}$ where d is the arithmetic mean of the ellipsoid diameters ($d = d_a = (A + B + C)/3$). The slip-flow correction only becomes significant for particles smaller than this range. In this model, oblate and prolate ellipsoids can have the same shape parameter, as for example the ellipsoids ($A = 1.0$, $B = 0.5$, $C = 0.5$) and ($A = 1.0$, $B = 0.9$, $C = 0.1$), both of which have $F = 0.5$.

The Ganser model is more sophisticated in that prolate and oblate ellipsoids can be distinguished using a second shape parameter. Ash3d uses a first shape parameter of $F = (B + C)/2A$, and an optional second $G = C/B$ which describes the ratio of the minor to intermediate axes of the ellipse. If G is not provided, it is assumed to be 1, meaning $B = C$ (prolate spheroids only). The drag coefficient for the Ganser model is given by

$$C_d = \frac{24}{\mathrm{Re}K_1}\left(1 + 0.1118\left(\mathrm{Re}K_1K_2\right)^{0.6567}\right) + \frac{0.4305K_2}{1 + \frac{3305}{\mathrm{Re}K_1K_2}} \tag{D.8}$$

where the Stokes and Newton shape factors (K_1 and K_2, respectively) are functions of the sphericity, ϕ, defined as the ratio of the surface area of a sphere with equivalent particle volume to the actual surface area of the particle.

$$K_1 = \left(\frac{1}{3} + \frac{2}{3}\phi^{-1/2}\right)^{-1} \tag{D.9}$$

$$K_2 = 10^{1{,}8148(-\log\phi)^{0.5743}} \tag{D.10}$$

Both Eq.s D.7 and D.8 have a first term that is a perturbation on the Stokes flow drag (Eq. D.2) which is the dominant term for low Re flows The second additive term in these drag models becomes dominant for high Re flows (Stokes and Newton regimes, respectively). If we normalize the ellipsoid diameters to the major axes, A, as $B = \beta A$ and $C = \gamma A$, then we can express these in terms of F and G as

$$\beta = \frac{2F}{1+G} \tag{D.11}$$

$$\gamma = \frac{2FG}{1+G} \tag{D.12}$$

We can then calculate the diameter of a sphere of equivalent volume as

$$d_v = \sqrt[3]{ABC} = A\sqrt[3]{\beta\gamma} \tag{D.13}$$

Then using the approximate formula for the surface area of an ellipsoid

$$S \sim 4\pi \sqrt[p]{\frac{(ab)^p + (ac)^p + (bc)^p}{3}} \tag{D.14}$$

where a, b, and c are the major, intermediate, and minor radii, and $p \approx 1.6075$ [**?**]. This results in a sphericity of

$$\phi = \frac{A_{sphere}}{A_{particle}} = (\beta\gamma)^{\frac{2}{3}} \left[\frac{\beta^p + \gamma^p + (\beta\gamma)^p}{3}\right]^{-\frac{1}{p}} \tag{D.15}$$

Particle diameter in the Ganser model is based on the equivalent volume sphere which is the geometric average of the particle diameters given in Eq. D.13.

As mentioned, most of these fall models require an iterative approach to solving for v_s. The exception is the Wilson and Huang model without the slip-correction, for which there is an explicit solution to a quadratic expression. Since the other fall models can be computationally expensive, the atmospheric parameters (ρ_a, η_a, λ_a) are only calculated on the nodes of the meteorological grid (not the computational grid). Fall velocities for each bin are calculated on these meteorological nodes, then interpolated to the computational grid. For this approximation to be valid, the density and viscosity of air must not be strongly non-linear between the meteorological grid nodes.

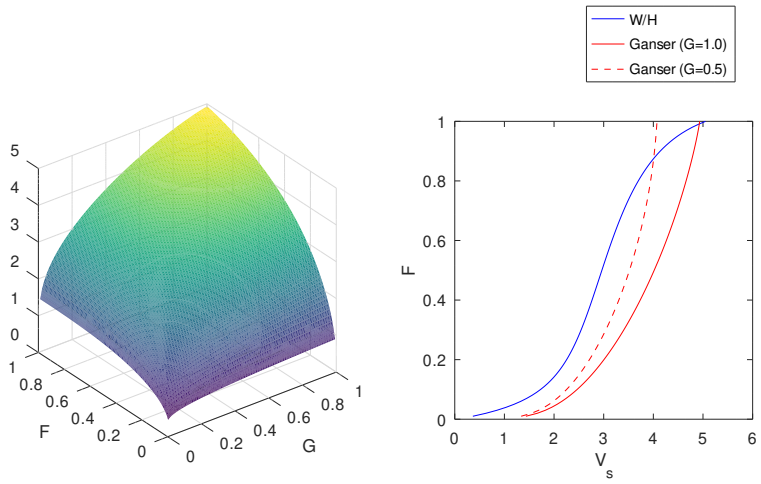


Figure D.1: Wilson/Huang and Ganser models of fall velocity for $d = 1\,\mathrm{mm}$ with atmospheric conditions at $5\,\mathrm{km}$ height.

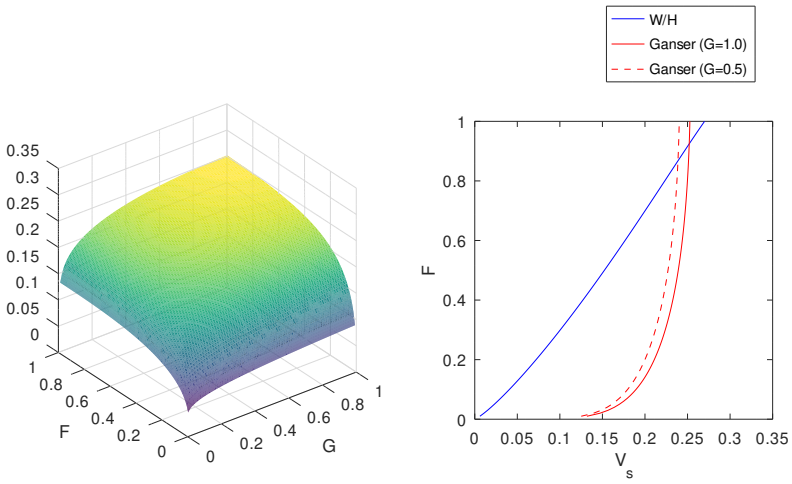


Figure D.2: Wilson/Huang and Ganser models of fall velocity for $d = 64\,\mu$m with atmospheric conditions at 5 km height.

Appendix E

Mathematical Terms

Table E.1: Mathematical Terms

x, y, z	Cartesian coordinate directions
λ,ϕ,z	Spherical coordinate directions (longitude, latitude, altitude)
q	Tephra concentration
q^*, q^{**},q^{***}	Tephra concentration at fractional steps
t	Time
$\mathbf{u}$,u, v, w	Wind velocity (vector and components)
$\mathbf{v_s}, v_s$	Fall velocity (vector and scalar)
$u', \bar{u}$	Perturbation, ensemble average of u
$\mathbf{K}$,K_x,K_y,K_z	Turbulent Diffusivity (tensor, components)
S	Source (emission) rate
d	Diameter of grain
ρ,ρ_d	Density of ash particle, deposit
a, b, c	Major, intermediate, minor axes of ellipsoid
F	Shape factor
ρ_a	Density of air
η_a	Dynamic viscosity of air
λ_a	Mean free path of an air molecule
g	Acceleration of gravity
C_d	Drag coefficient
C_c	Cunningham slip-flow correction factor
α, β, γ	Slip-flow parameters
Kn	Knudsen number
Re	Reynolds number
M_B	Molecular weight of dry air
R	Ideal gas constant
P, P_0	Pressure, Pressure at ground surface
T, T_0	Temperature, Temperature at ground surface
δ	Scale height
θ	Adiabatic lapse rate
k	Suzuki parameter
H	Plume height
V	Eruptive volume
$Q^n_{i,j,k}$	Average concentration in cell i, j, k at time n
$\Delta\kappa$	Volume of cell (capacity)
$F^n_{i-1/2,j,k}$	Flux across interface $i-1/2$ at time n
$\sigma_{i-1/2,j,k}$	Area of interface of cell i, j, k
$\mathcal{A}^{\pm}$	Fluctuation
$\tilde{F}_{i-1/2,j,k}$	High-order correction to fluctuation
$\widetilde{\Delta Q}_{i-1/2,j}$	Limited wave at $i-1/2$
ω	Rotational velocity
λ_0, ϕ_0	Pole of rotation
L_1	Error norm
U_0, V_0, W_0	Characteristic velocity in x, y, z
Z_0	Height of tropopause
λ_{xy}	Horizontal length scale
ζ, ζ_0	Transient length scale, characteristic length scale
Q_0	Characteristic initial concentration

Appendix F

Test Cases and convergence tests

We have implemented many different levels of testing for Ash3d. The simplest set of tests is the quick runs invoked via `make check` when building the software. These tests are intended to run in just a few minutes with pass or fail results. The second set of tests cases are the verification tests which ensure that Ash3d is correctly solving the equations that we intend. Validation tests are provided in the examples folder of the `volcano-ash3d` repository. Lastly, the windfile testing suite ensures that Ash3d correctly interfaces with the variety of NWP data products available through the MetReader library, along with the different formats for those products.

F.1 Difference tests: make check

To test that a build of Ash3d is calculating results correctly, a quick check is to run `make check` after compiling the code. This makefile target runs the script `run_tests.sh` which in turn runs a sequence of four testing scripts in directories `tests/test_[1-4]`.

F.1.1 test_01: Cartesian, Advection

This test case uses a simple Cartesian grid with a Suzuki source (k=4) with a single tracer grainsize. The wind data used is just an artificial 1-D profile with a spriral windfield.

```
Hanford
0    5   5  1 2 3 0 5  #wind time, number of levels, nvar ivars(1:nvar)
-2062.26 2606.69 0 4 265.0 25.0 25.0 25.0 6371.229 # (-120.0   46.20) in LCC grid 212 coords
0     -4.0   0.0 1000.0  10.0
5000   0.0   4.0  560.0 -5.5
10000  4.0   0.0  262.0 -47.3
15000  0.0  -4.0  126.0 -58.1
20000 -4.0   0.0   58.0 -54.9
```

Output files are ESRI ASCII data files for cloud concentration, cloud height, and cloud load (all at 2 and 4 hours), and the cloud arrival time. These output files are tested against the version stored in `test_01`

`output` using the tool `Ash3d_ASCII_check`. This tool calculates the L2 norm of the difference between two ESRI ASCII files and reports PASS or FAIL based on a provided tolerance.

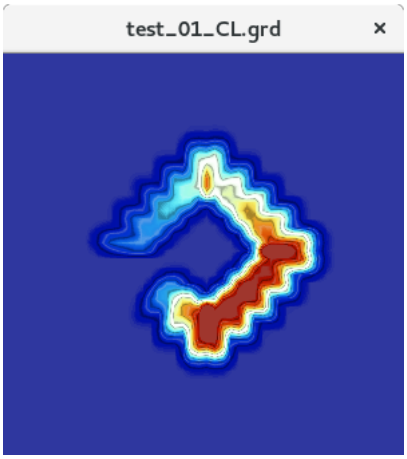


Figure F.1: Cloud Load for simple test case of advection for test 1.

The output files are not tested for equivalance since there are many reasons there might be slight differences in either build-time or run-time values, such as: CFL value, minimum/maximum `dt`, floating point precission, Vz approximation, etc.

F.1.2 test_02: Cartesian, Diffusion

This test case uses essentially the same input file as in test case 1, but a uniform wind of y=1 m/s and a diffusivity of 500 m2/s. Stored output files are those calculated using the Crank-Nicolson scheme. While both the Crank-Nicolson and the explicit scheme converge to the same solution, at the grid resolution of this test case, the solutions are notably different.

F.1.3 test_03: Cartesian, Source/Fall-model/vertical grid

This set of tests also uses the input file from `test_01` as a template, but varies different settings for the source term, the fall model and the structure of the vertical grid. All these subcases use the simple ASCII spiral windfield.

F.1.4 test_04: Global grid, realistic atmosphere

Test 4 uses the NCEP 50-year reanalysis data for the year 1980. These are the same windfiles that are needed for running the tests for `volcano-ash3d-metreader`. If you do not already have these data installed from during the MetReader installation, you can download them using a script installed with MetReader:
`/opt/USGS/bin/autorun_scripts/get_NCEP_50YearReanalysis.sh 1980`
which will copy the data to:
/texttt/data/WindFiles/NCEP/1980/[air,hgt,omega,uwnd,vwnd].1980.nc

Subcase	Source Type	Fall Model	Vert. grid
0	Suzuki	Wilson-Huang	constant dz
1	Line source	Wilson-Huang	constant dz
2	point source	No fall (tracer)	constant dz
3	point source	Wilson-Huang	constant dz
4	point source	Ganser	constant dz
5	point source	Stokes+slip	constant dz
6	profile source	Wilson-Huang	constant dz
7	line source	Wilson-Huang	constant dz
8	point source	Wilson-Huang	piecewise linear dz
9	point source	Wilson-Huang	constant log(dz)
10	point source	Wilson-Huang	custom dz

Table F.1: Sub-cases for TestCase 3

To run this set of sub-cases, Ash3d must have been compiled with the netcdf libraries. These test cases take significantly longer than those of Test Case 1-3, but should still complete in a few minutes on modern hardware.

Sub-cases 0-3 all have the same grid and different approximations of the same event. Sub-case 0 uses a standard Suzuki (k=4) source profile with a single grain-size bin (10 μm) that represents the distal cloud fraction. Sub-case 1 is equivalent to 0, but with the `umbrella_air` source used. Sub-case 2 and 3 use the Suzuki and umbrellla source terms, but with a full 12-bin grain-size distribution that is used to model typical deposits, Since the ash cloud runs assume the distal ash is 5% of the erupted volume, the eruptive volume for sub-case 2 and 3 are 20 times that of sub-cases 0 amd 1. Note, the umbrella source terms are currently only availablee with longitude/latitude grids. Sub-case 4 is the same as 0, but with a global grid, testing the periodic boundary conditions as the drifting ash cloud wraps around the grid.

Subcase	Source Type	E.Vol	GSD	Domain
0	Suzuki (k=4)	5.000E-2	1 gs (10 μm)	LL 37 x 20.5
1	Umbrella_air	5.000E-2	1 gs (10 μm)	LL 37 x 20.5
2	Suzuki (k=4)	1.000E+0	12-gs (deposit)	LL 37 x 20.5
3	Umbrella	1.000E+0	12-gs (deposit)	LL 37 x 20.5
4	Suzuki (k=4)	5.000E-2	1 gs (10 μm)	LL global/periodic

Table F.2: Sub-cases for TestCase 4: SC0 = Suzuki ash cloud; SC1 = Umbrella ash cloud; SC2 = Suzuki deposit; SC3 = Umbrella deposit; SC4 = Suzuki ash cloud global

Sub-cases 5-7 test the different implementations of topography. SC5 uses a Suzuki source with a Cartesian grid with no topography, and a 1-D rotating windfield. The source extends from the base of the z-grid (z=0) to 15 km. SC6 is the same configuration, but with a constant topography of $Z_s = 1$ and using `ZScaling_ID`=1 (shifted z-grid). The vent elevation is set to $Z_v = 1$ and the plume height is set to 16 km, 15 km above the vent. A different windfile is

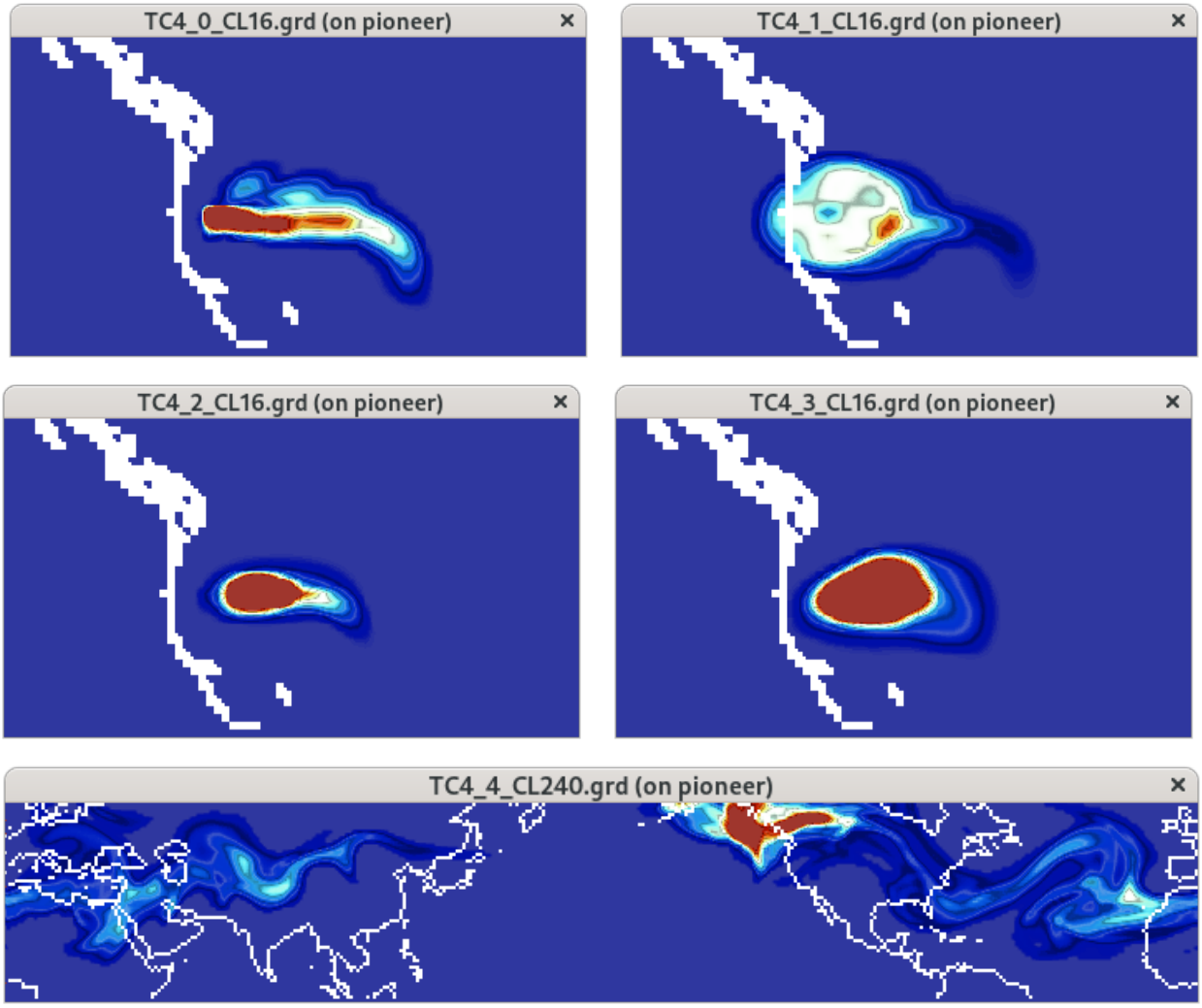


Figure F.2: Cloud Load for simple test case of advection for test 4.

used with values shifted up by 1 km. With this configuration, SC6 should produce equivalent deposit calculations as SC5.

SC7 has the same source geometry as SC6, but uses `ZScaling_ID`=2 (scaled z-grid). The scaled vertical grid uses $s = \frac{z - z_s}{z_{top} - z_s}$. Moreover, SC7 uses a vertical grid spacing that is logarithmic.

F.2 Unit testing of functions

F.2.1 Atmosphere.f90

- `Dens_IdealGasLaw`
- `Visc_Sutherland`
- `lambda_MeanFreePath`

F.2.2 Tephra.f90

- `vset_WH`
- `vset_WH_slip`
- `vset_WH_PCM`
- `vset_Gans`
- `vset_Gans_slip`
- `vset_Stokes_slip`

F.3 Convergence Tests

The advection and diffusion routines that are implemented in Ash3d are 2nd order routines when high-resolution, flux-limiters are employed. To test that the implementation of these routines is correct and that they converge at the expected rate, we have a series of convergence test cases that can be run as an optional module. The repository with this module is available at:
https://github.com/hschwaiger-usgs/volcano-ash3d-optionalmodules

To build this module, just make sure that the variable in the makefile pointing to the core-code source files is correct. E.g.
`ASH3dCCSRC= /work/USGS/Software/GIT/volcano-ash3d/src` This repository has a top-level file (`Ash3d_TC.F90`) that replaces the `Ash3d.F90` from the core-code, and links to the optional module `src/Optional_Modules/TestCases/Testcases.f90`. This module set initial source conditions and wind velocities for 6 test cases, each with several sub-cases. There are also routines for calculating errors between model results and true solutions for this test problems. The tests cases (and sub-cases) are determined at compile-time via proprocessor flags with the whole testing suite managed by the script `examples/Testcases/run_all.sh`. In this script, there are some variables near the start of the file that control the script. The number of limiters to test is set with:

```
il=1 # 0-6: 0 is just no-limiter, 1 is NO and Superbee, 6 is all
```

The highest resolution is set with:

```
ix=3 # 1-5: 5 is the max prepared but that takes overnight
```

And the specific cases to run is set with:

```
# Specify which cases to turn off, by setting the corresponding value to 0
    #  1 2 3 4 5 6
cases=(0 0 0 0 1 0)
```

In this example, only test case 5 would be run. `cases=(1 1 1 1 1 1)` runs all 6 test cases.

F.3.1 Test case 1: Horizontal advection of smooth source

The simplest test case isolates the horizontal advection routines and tests for the accuracy of advection of a smooth initial concentration profile from the origin along each coordinate direction (SC1:$x+$,$y=0$; SC2:$x-$,$y=0$; SC3:$x=0$,$y+$; SC4:$x=0$,$y-$; SC5:$x+$,$y+$; SC6:$x-$,$y-$; SC7:$x-$,$y+$; SC8:$x+$,$y-$). Output is written once when the advected pulse is still entirely in the domain and a second time when the pulse has partially the domain. The geometry and an example of SC1 is shown in Figure F.3.

The test is initially conducted on a Cartesian grid, then repeated on a longitude/latitide grid. The windfield is homogeneous and constant for the Cartesian case, but heterogeneous in the longitude/latitude case, corresponding to rigid-body rotation on a sphere.

F.3.2 Test case 2: Vertical advection of smooth source

Test case 2 isolates the vertical advection routines and uses the same set-up as in Test case 1. There are four sub-cases (SC1:v_z+,$V_f=0$; SC2:v_z-,$v_f=0$; SC3:$v_z=0$,v_f+; SC4:$v_z=0$,v_f-). These tests are only for a Cartesian grid.

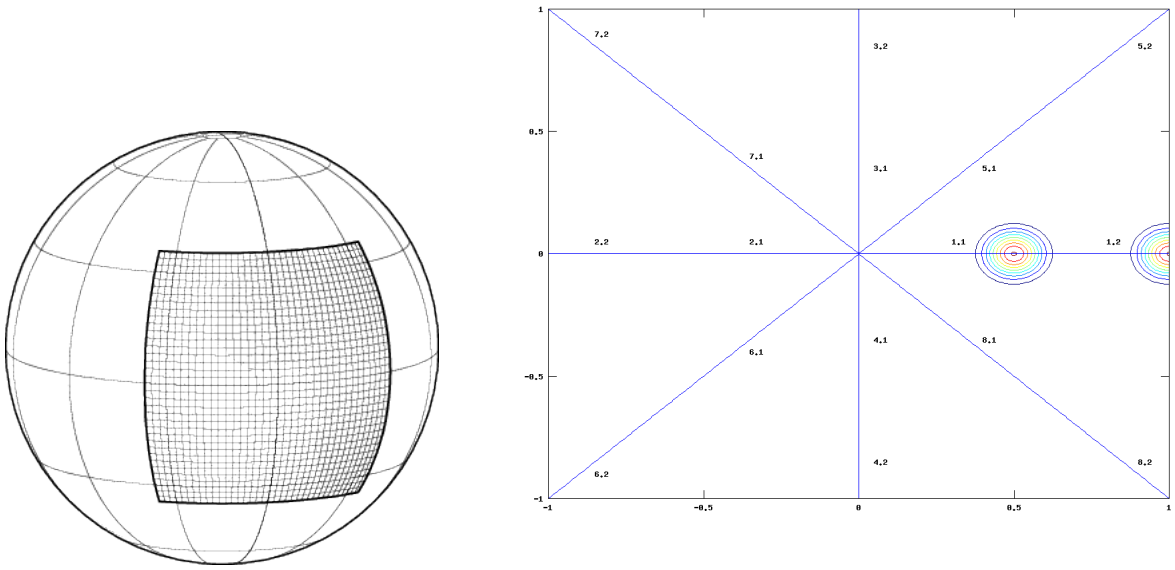


Figure F.3: Grid and example advection for Test Case 1.1

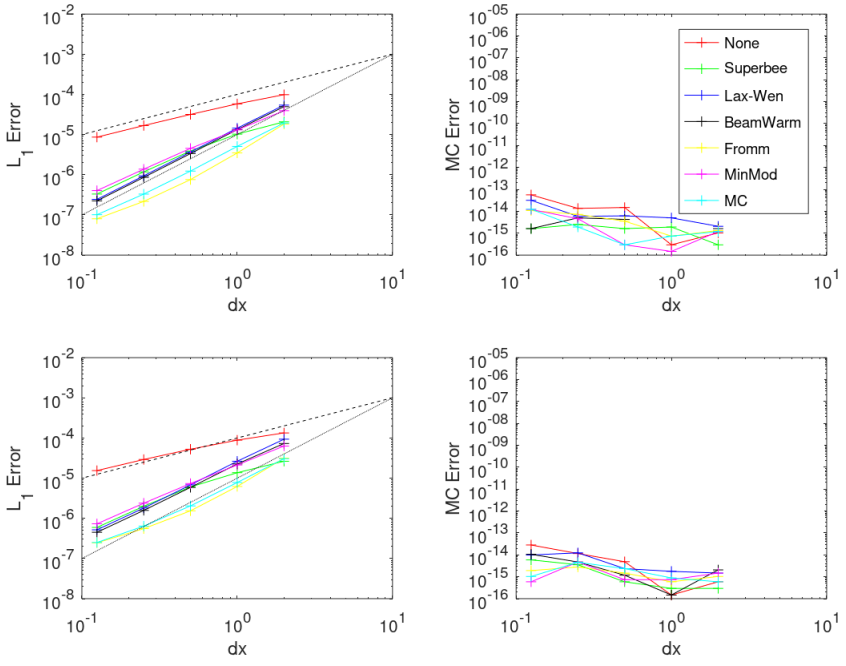


Figure F.4: Convergence plot for TC1.5. The top row is for the mid-point; bottom row is for the boundary. Left column is the L_1 error; right column is the mass concervation error.

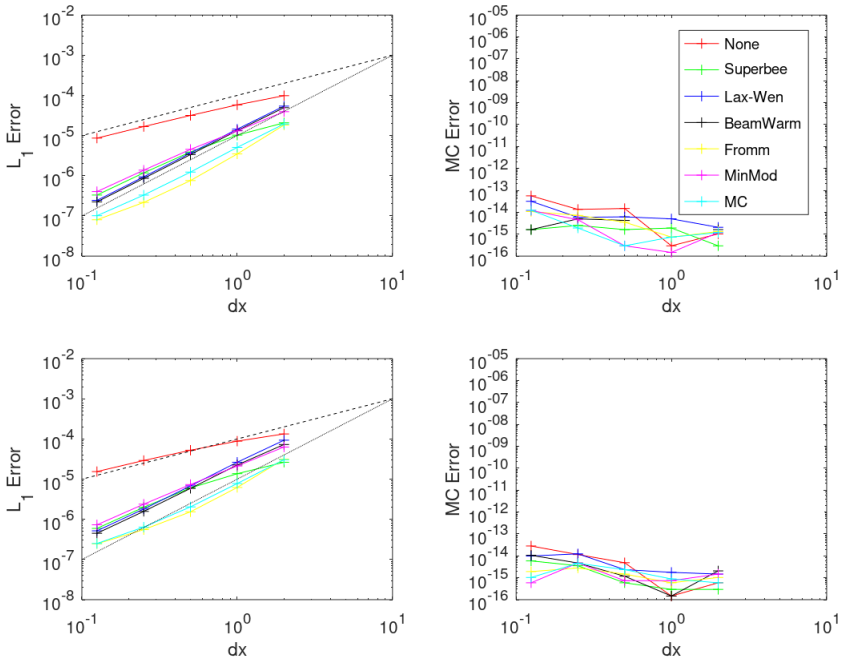


Figure F.5: Convergence plot for TC2.1.

F.3.3 Test case 3: Horizontal rotational advection of 2-D cone and box

This test case examines the effect of rigid-body rotation of a concentration profile through one revolution. The concentration profile is given below for a Cartesian grid and consists of a cone centered at $x = -0.45$, $y = 0$ and a box located at $0.1 < x < 0.6$, $-0.25 < y < 0.25$.

$$q_0 = \begin{cases} 1 & : \text{if } 0.1 < x < 0.6 \text{ and} \\ & \quad -0.25 < y < 0.25 \\ 1 - r/0.35 & : \text{if } r < 0.35 \\ 0 & : \text{Otherwise} \end{cases} \tag{F.1}$$

where $r \equiv \sqrt{(x + 0.45)^2 + y^2}$. The wind field is given by

$$u(x, y) = 2y \qquad v(x, y) = -2x \tag{F.2}$$

Figure F.1 shows that the DCU implementation of advecting in x, then advecting in y (switching the order with each time step), introduces a splitting error and reduces the order of accuracy to 1rst order.

F.3.4 Test case 4: 1-D diffusion

This test case checks the implementation of the diffusion routines along the three coordinate axes. The model consists of diffusion with a discontinuous initial concentration and with a

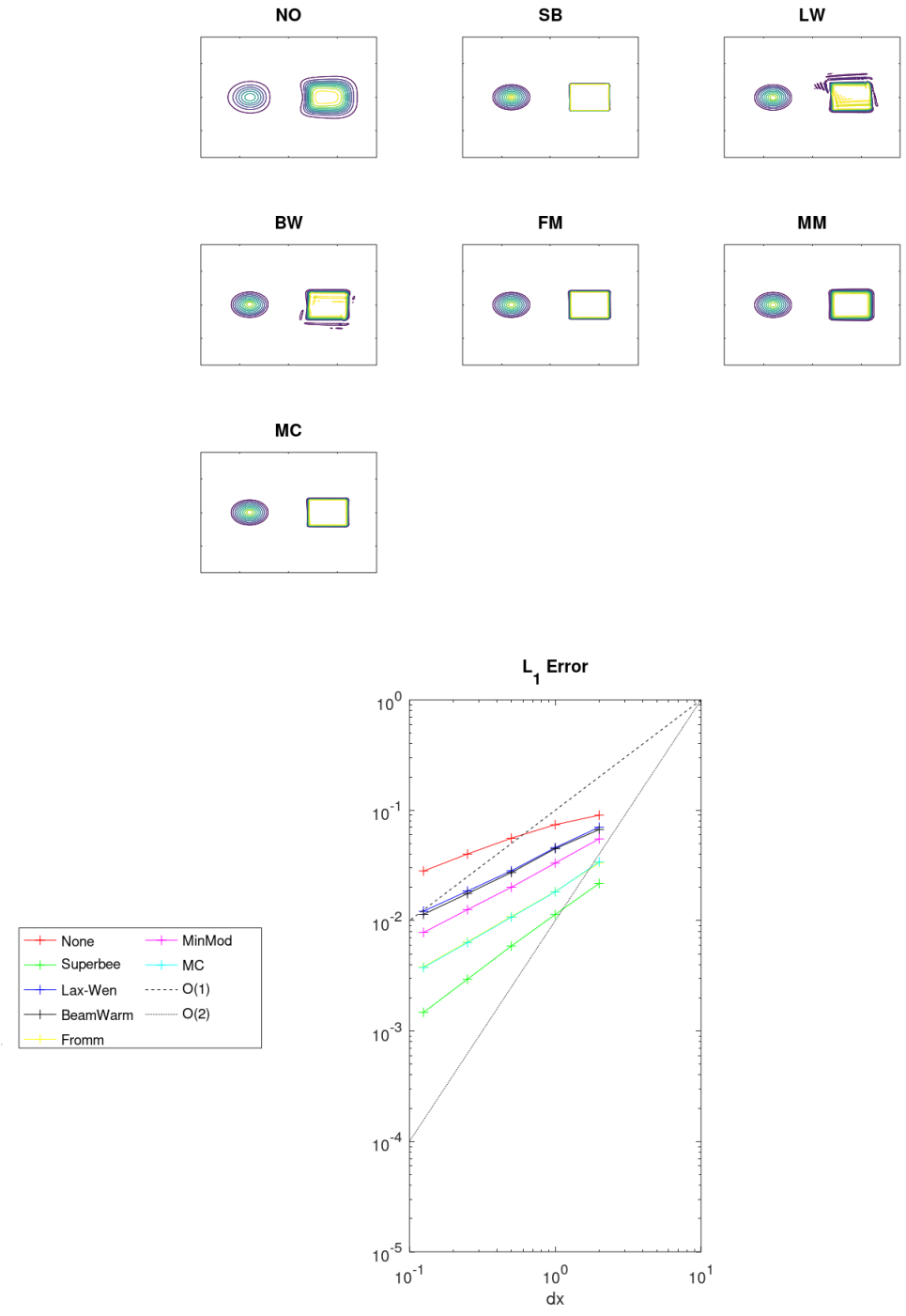


Figure F.6: Solution plot for TC3 and convergence behaviour.

diffusivity contrast at the origin. There are 6 sub-cases for this test for diffusion along x, y, and z with the explicit solver, then repeating x, y, and z with the Crank-Nicolson solver.

The current implementation of diffusion is 2nd order for homogeneous diffusivities, but is reduced to 1rst order for heterogeneous diffusivities.

F.3.5 Test case 5: Reversible horizontal shearing

To evaluate the ability of these numerical schemes to track advection in highly deforming wind fields, a test was run using a transient, rotational wind field where the rotational velocity about the pole ($\phi_0 = 0°$, $\lambda_0 = 0°$) is given by (shown in figure b)

$$\omega(\phi, \lambda) = -0.5 \sin(d\pi/18) \cos(\pi t/4) \tag{F.3}$$

where d is the angular distance (in degrees) of ϕ, λ from ϕ_0, λ_0. The initial concentration distribution (shown in figure a) is given by

$$q(\lambda, \phi, z, t = 0) = 0.5(1.0 + \cos(\pi r)) \tag{F.4}$$

where r is given by:

$$r = \sqrt{(\phi + 16.0)^2 + \lambda^2/16} \tag{F.5}$$

$$r = \min(1.0, r) \tag{F.6}$$

The initial concentration distribution is deformed into a long strand at $t = 2$ (figure c), then the deformation reverses back to the initial configuration. A similar test in Cartesian coordinates was also conducted. Versions of the test cases for both Cartesian and spherical coordinates are shown.

Figure F.8 shows the behavior of the various numerical methods for this test problem at $t = 4$ when the deformation should return to zero. Each plot of figure should be compared with figure a. Since diffusion is not calculated in this test problem, any distortion of the final profile is a result of numerical diffusion. The CTU method without the use of a limiter recovers the initial profile with the greatest fidelity. However it is possible in this case, since the use of limiters is non-linear, that errors in the advection in the clock-wise rotation are simply reversed in the counter-clock-wise phase of the oscillation.

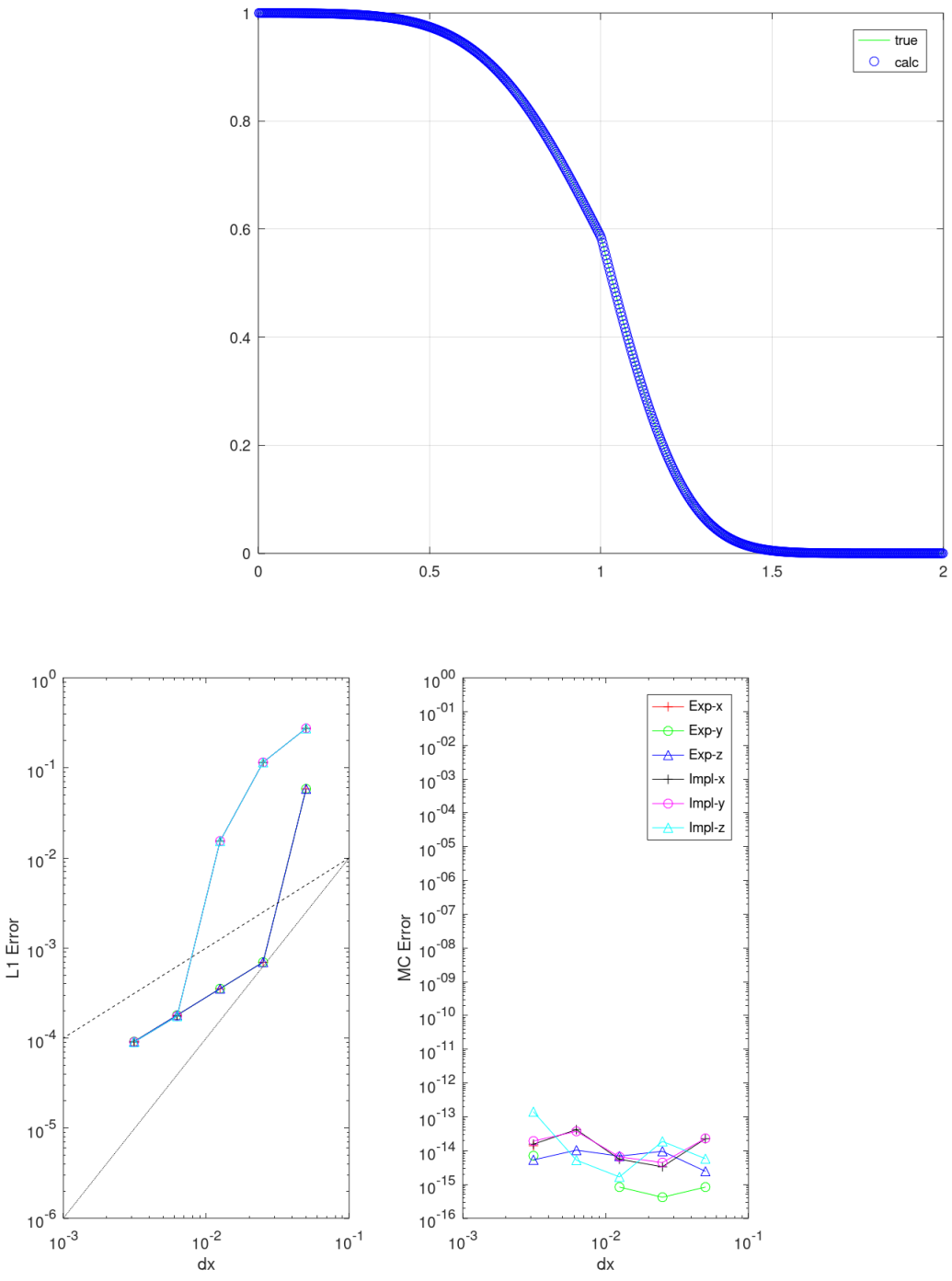


Figure F.7: Solution plot for TC4 and convergence behaviour.

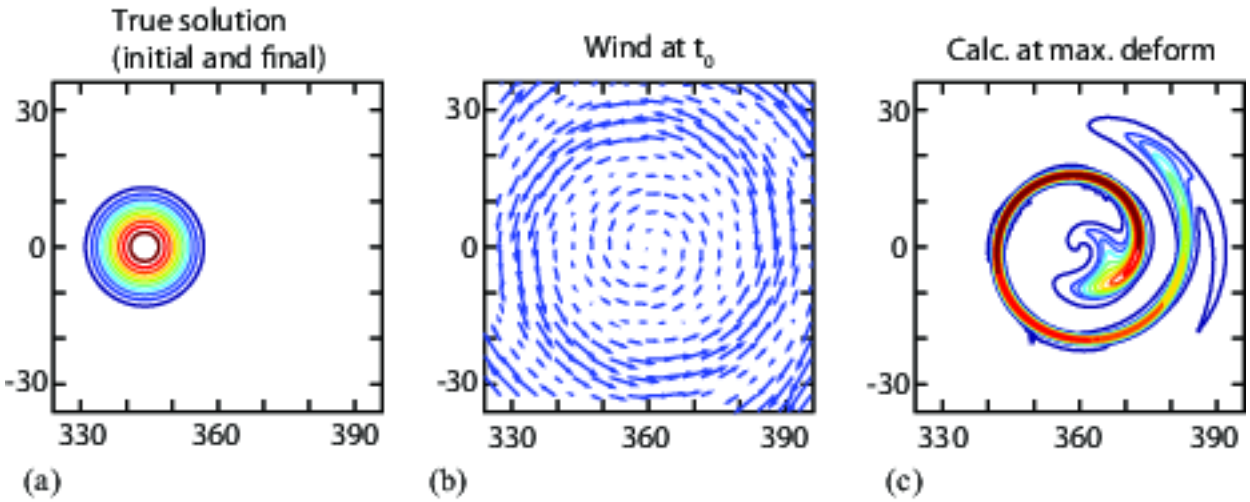


Figure F.8: Initial/final solution for TC5 along with peak deformation.

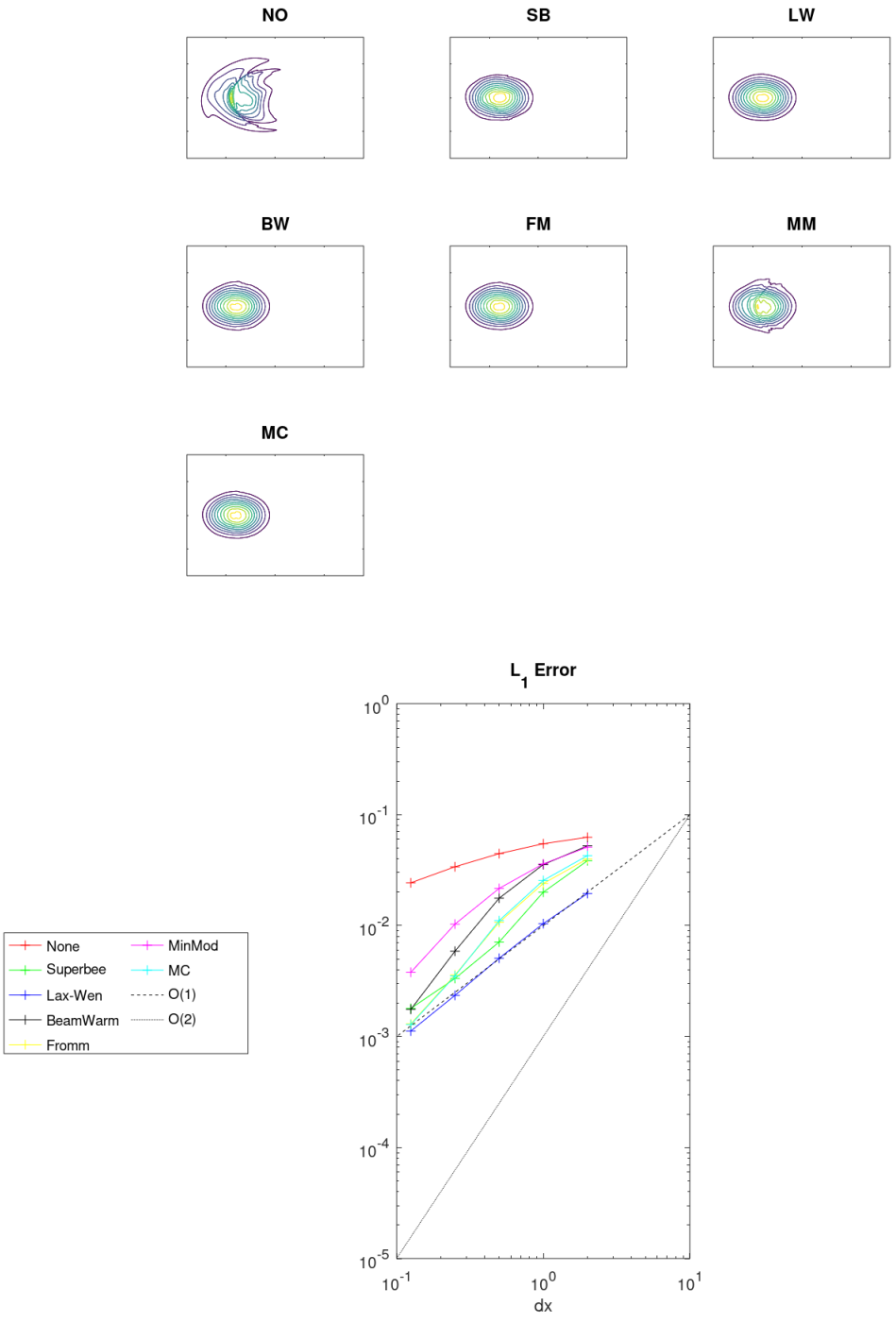


Figure F.9: Solution plot for TC5 and convergence behaviour.

F.3.6 Test case 6: Method of manufactured solutions

In addition to the two test cases shown above (rotational advection and reversible rotational shear) simple 1-dimensional advection and 1-dimensional diffusion tests were conducted along the coordinate axis to verify convergence properties. Simple tests such as these are useful for identifying obvious bugs in the code; however, they do not fully test all aspects of the algorithm. In general, analytic solutions are not available for the advection-diffusion-sedimentation equation in a non-trivial wind field with realistic pressure and temperature profiles (See [**?**] for solutions in simplified atmospheric conditions.). However, the Method of Manufactured Solutions (MMS) provides some guidance on how to generate analytic solutions for non-trivial atmospheric conditions which will invoke all components of the algorithm, all terms of the governing equation and boundary conditions. This will allow us to confirm the order of accuracy of the complete algorithm.

If a domain is specified and atmospheric and boundary conditions prescribed, then a "source" (a transient spatial distribution of tephra characterizing the eruption) results in a "solution" (a transient spatial distribution of airborne and deposited tephra) through the governing equation. In Eq. 1.5, the source is S and the solution is q. In general, we do not know the solution *a priori* that is the result of the source term and boundary conditions. The idea behind the Method of Manufactured Solutions is to find a compatible set of functions for the source and the solution by specifying q and solving for S.

To employ the MMS method, we first choose a model geometry (domain and topography) and define the physical state (p, T, u, v, w, $\mathbf{K}$). Next, we simply choose a non-trivial solution, q, ideally one with a 3-dimensional, transient structure that is easily differentiable. This artificial solution need not be physically realistic since our goal is to evaluate the performance of the algorithm, not model a particular physical process. This solution can be inserted into the governing equation (Eq. 1.5) and evaluated to generate a non-trivial, three-dimensional, transient source term, S. Note that this source term will not resemble the vertical distribution described in section **??**, but instead will be a transient three-dimensional function. Initial conditions are generated by simply evaluating the chosen solution at $t = t_0$. Boundary conditions can be determined by evaluating q along the boundary. Applying the algorithm with these initial and boundary conditions, along with the source term, will generate a numerical solution that can be compared with the solution that was initially chosen. Grid convergence studies of this test case will quantify the order of accuracy of the full algorithm.

We use the following domain for the MMS test. $-100 < x < 100\,\text{km}$, $-100 < y < 100\,\text{km}$, $0 < z < 20\,\text{km}$. We use a standard atmosphere for the pressure and temperature profiles given by : $P = P_0 \exp(-z/\delta)$ and $T = T_0 + \theta z$. For the wind velocity structure, we adopt a horizontal model to mimic wind shear with small vertical perturbations: $u = U_0$, $v = V_0/2\,(1 + \tanh(z - Z_0))$, $w = -W_0 \cos(\pi x/\lambda_{xy}) \cos(\pi y/\lambda_{xy})$. The diffusivity is set to a constant. The fall model is set via the equations of section **??** using the ideal gas law and Sutherland's Law to calculate ρ_a and η_a.

The solution we choose is given below.

$$q = Q_0 \,\text{sech}\left(\frac{x}{\lambda_{xy}}\right) \text{sech}\left(\frac{y}{\lambda_{xy}}\right) \text{sech}\left(\frac{z}{\zeta}\right) \tag{F.7}$$

where $\zeta = W_0 t + \zeta_0$. To calculate the source term, S, we need to calculate the first partial

derivatives of q with respect to t, and both first and second partial derivatives with respect to the spatial coordinates.

$$\frac{\partial q}{\partial t} = q\left[z\frac{W_0}{\zeta^2}\tanh(z/\zeta)\right] \tag{F.8}$$

$$\frac{\partial q}{\partial x} = q\left[-\frac{\tanh(x/\lambda_{xy})}{\lambda_{xy}}\right] \tag{F.9}$$

$$\frac{\partial^2 q}{\partial x^2} = q\left[\frac{\tanh^2(x/\lambda_{xy})}{\lambda_{xy}^2} - \frac{\mathrm{sech}^2(x/\lambda_{xy})}{\lambda_{xy}^2}\right] \tag{F.10}$$

$$\frac{\partial q}{\partial z} = q\left[-\frac{\tanh(z/\zeta)}{\zeta}\right] \tag{F.11}$$

$$\frac{\partial^2 q}{\partial z^2} = q\left[\frac{1}{\zeta^2}\left(\tanh^2(z/\zeta) - \mathrm{sech}^2(z/\zeta)\right)\right] \tag{F.12}$$

The equations for first and second derivatives in y are of the same form as Eq.s F.9 and F.10. The choice of Eq. F.7 is not unique, but is guided by the simple form of the derivatives in Eq.s F.8–F.12. An arbitrarily complex form of q can be chosen, as determining S is straight-forward once the derivatives are calculated; although it may be more cumbersome to implement. Assembling these terms into the governing equation (Eq. 1.5) leads to a simple expression for the source term, S, which is a continuous function of space and time. By selecting individual constants (i.e. U_0, V_0, W_0, Z_0, ζ_0) we can selectively activate or deactivate branches of the algorithm. If all are activated, we have an analytic solution against which we can compare the full advection-diffusion-sedimentation equation. This is particularly important for calculating splitting errors due to the decomposition of the governing equation via the method of fractional steps. The constants used in the testcase are given in Table **??**.

L_1-norm convergence results are shown in figure **??**. The full simulation using the semi-Lagrangian scheme for the horizontal advection produce results that converge with an order accuracy of 1.5. Without the use of limiters, the DCU and CTU schemes converge with second-order accuracy. With minmod, superbee and MC limiters, the accuracy of the calculations with DCU and CTU for horizontal advection reduces to approximately order 1.8. The fact that these convergence rates are nearly second-order suggests that splitting errors are minimal. Although the use of limiters in this particular case reduces the order of convergence slightly, they significantly improve the resolution of sharp boundaries.

F.4 Validation Testing

F.4.1 Spurr (Deposit): August 19, 1992

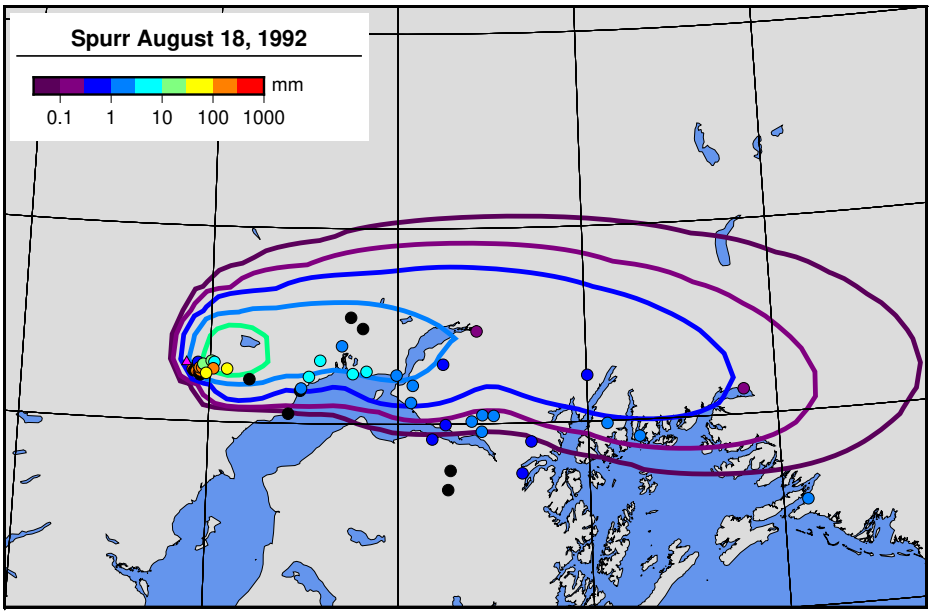


Figure F.10: Deposit

F.4.2 Mt. St. Helens (Deposit with Aggregates): May 18, 1980

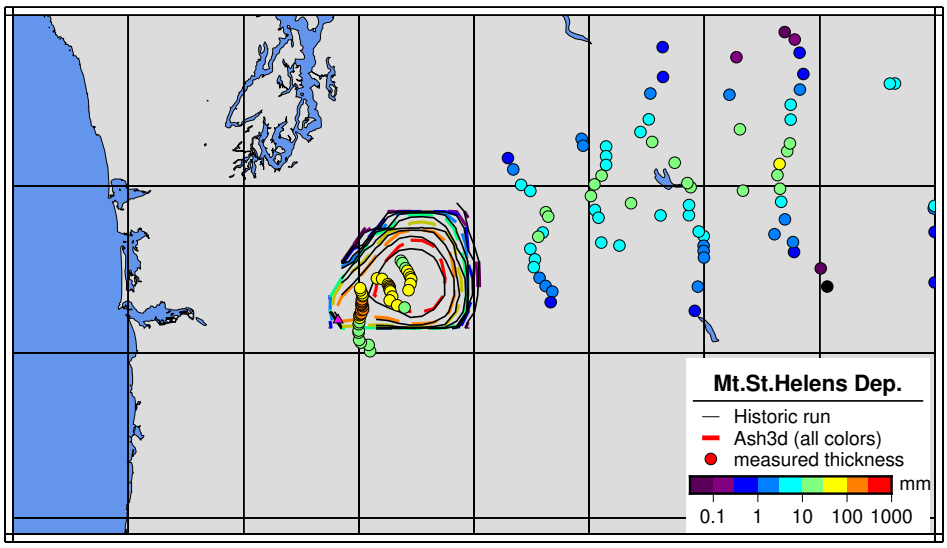


Figure F.11: Deposit

F.4.3 Kasatochi (Ash cloud load): August 8, 2008

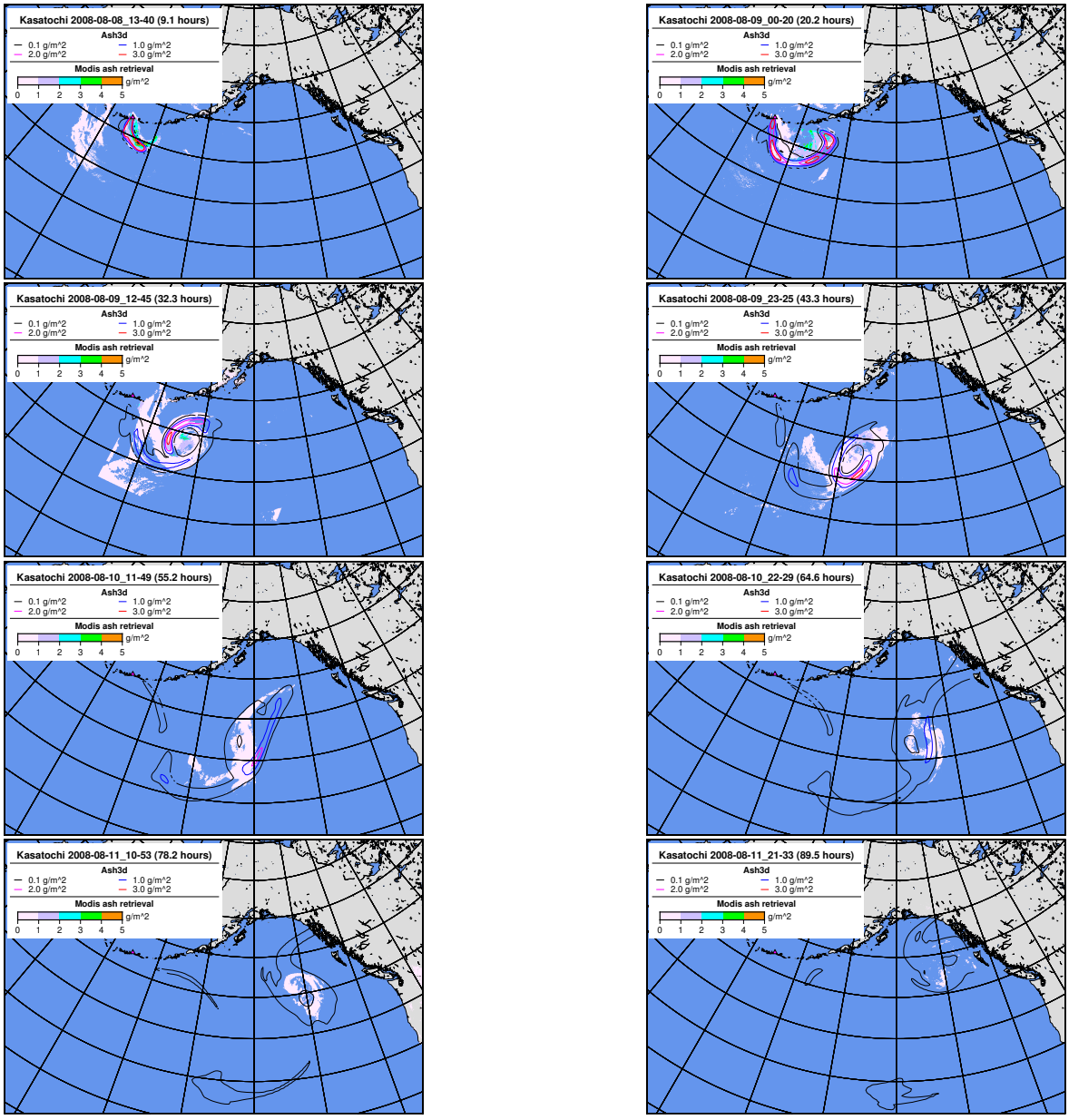


Figure F.12: Cloud load

F.4.4 Mazama (Deposit from Umbrella Source)

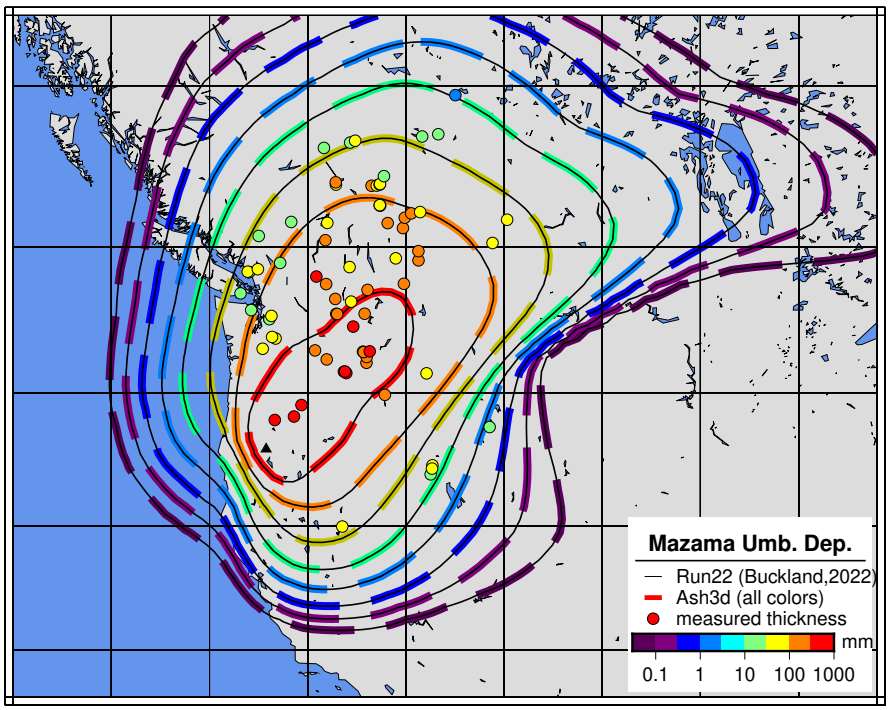


Figure F.13: Deposit

F.4.5 Kelud (Ash Cloud from Umbrella Source), Feb. 13, 2014

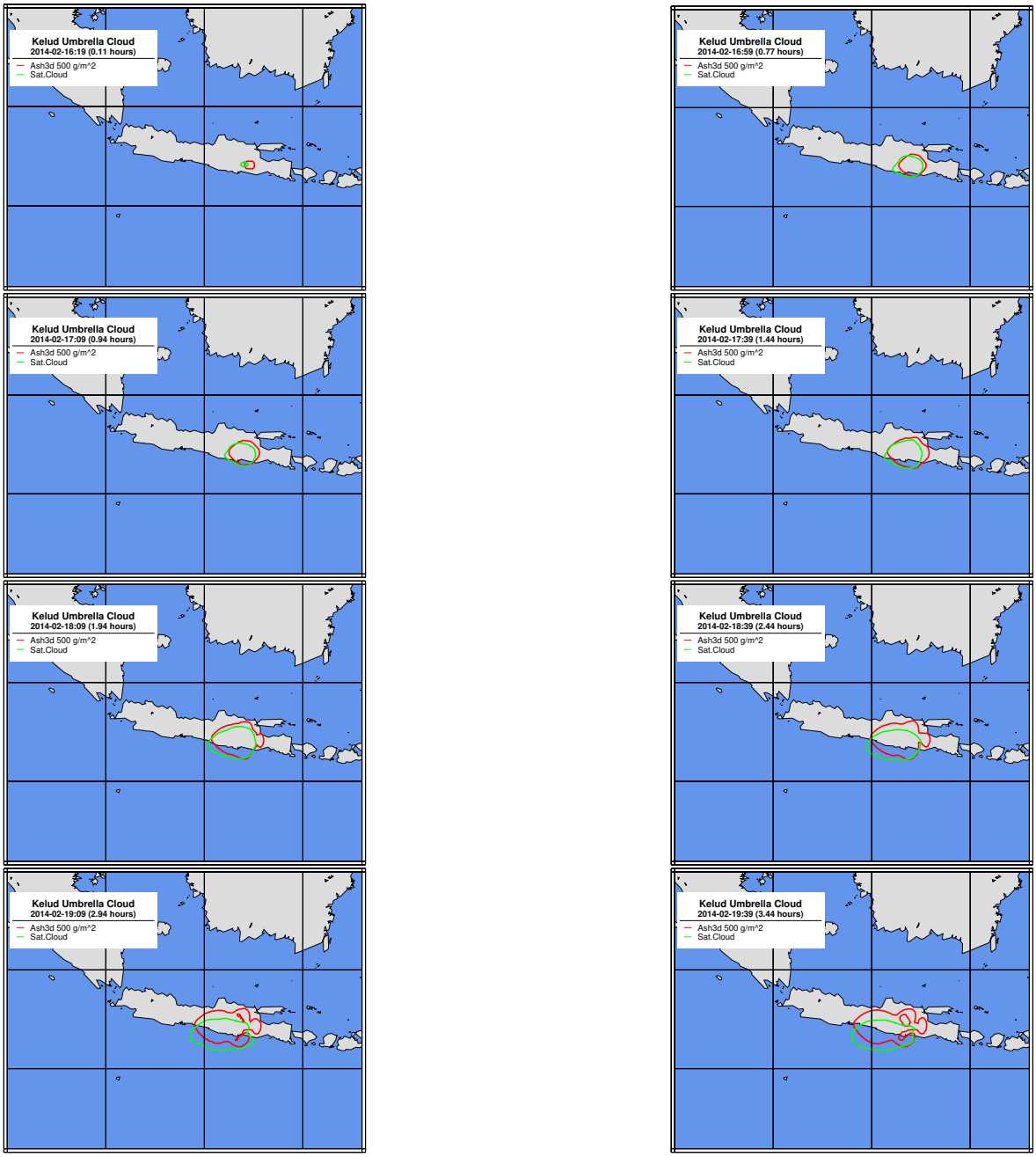


Figure F.14: Umbrella Cloud Spreading rate

Bibliography

[1] Barker, S. J., Eaton, A. R. V., Mastin, L. G., Wilson, C. J. N., Thompson, M. A., Wilson, T. M., Davis, C., and Renwick, J. A. Modeling ash dispersal from future eruptions of Taupo supervolcano. *Geochemistry, Geophysics, Geosystems 20*, 7 (2019), 3375–3401.

[2] Bird, R., Stewart, W. E., and Lightfoot, E. N. *Transport Phenomena*. John Wiley and Sons, 1960.

[3] Carey, S. N. Modeling of tephra fallout from explosive eruptions. In *Monitoring and mitigation of volcano hazards*, R. Scarpa and R. I. Tilling, Eds. Springer Berlin/Heidelberg, 1996, pp. 429–461.

[4] Costa, A., Smith, V. C., Macedonio, G., and Matthews, N. E. The magnitude and impact of the youngest toba tuff super-eruption. *Frontiers in Earth Science 2* (2014).

[5] Denlinger, R. P., Pavolonis, M., and Sieglaff, J. A robust method to forecast volcanic ash clouds. *Journal of Geophysical Research 117*, D13 (2012).

[6] Dunbar, N. W., Iverson, N. A., Eaton, A. R. V., Sigl, M., Alloway, B. V., Kurbatov, A. V., Mastin, L. G., McConnell, J. R., and Wilson, C. J. N. New Zealand supereruption provides time marker for the Last Glacial Maximum in Antarctica. *Antarctica. Sci Rep. 7*, 1 (2017).

[7] Ganser, G. H. A rational approach to drag prediction of spherical and nonspherical particles. *Powder Technology 77*, 2 (1993), 143–152.

[8] Harris, A., Gurioli, L., Hughes, E., and Lagreulet, S. Impact of the eyjafjallajökull ash cloud: A newspaper perspectiv. *Journal of Geophysical Research 117* (2012).

[9] Jacobson, M. Z. *Fundamentals of Atmospheric Modeling*, 2 ed. Cambridge University Press, 2005.

[10] LeVeque, R. J. *Finite Volume Methods for Hyperbolic Problems*. Cambridge University Press, 2003.

[11] Mastin, L., Guffanti, M., Ewert, J., and Spiegel, J. Preliminary spreadsheet of eruption source parameters for volcanoes of the world. Open-file Report 2009-1133, U.S. Geological Survey, 2009.

[12] Mastin, L., Guffanti, M., Servranckx, R., Webley, P., Barsotti, S., Dean, K., Durant, A., Ewert, J., Neri, A., Rose, W., Schneider, D., Siebert, L., Stunder, B., Swanson, G., Tupper, A., Volentik, A., and Waythomas, C. A multidisciplinary effort to assign realistic source parameters to models of volcanic ash-cloud transport and dispersion during eruptions. *Journal of Volcanology and Geothermal Research 186*, 1 (2009), 10–21.

[13] Mastin, L., Randall, M. J., Schwaiger, H. F., and Denlinger, R. P. Users guide and reference to the web interface of ash3da three-dimensional model for atmospheric tephra transport and deposition. Open-file Report 2013-1122, U.S. Geological Survey, 2013.

[14] MAstin, L. F., Eaton, A. R. V., and Lowenstern, J. B. Modeling ash fall distribution from a yellowstone supereruption. *Geochemistry, Geophysics, Geosystems 15*, 8 (2014), 3459–3475.

[15] Mastin, L. G., and Eaton, A. R. V. Comparing simulations of umbrella-cloud growth and ash transport with observations from pinatubo, kelud, and calbuco volcanoes. *Atmosphere 11*, 10 (2020).

[16] Mastin, L. G., Eaton, A. R. V., and Durant, A. J. Adjusting particle-size distributions to account for aggregation in tephra-deposit model forecasts. *Journal of Volcanology and Geothermal Research 16*, 14 (2020), 9399–9420.

[17] Mastin, L. G., Schwaiger, H., Schneider, D., Wallace, K., Schaefer, J., and Denlinger, R. Injection, transport, and deposition of ash during event 5 at Mount Redoubt, March 23, 2009. *Journal of Volcanology and Geothermal Research 259* (2013), 201–213.

[18] Pfeiffer, T., Costa, A., and Macedonio, G. A model for the numerical simulation of tephra fall deposits. *Journal of Volcanology and Geothermal Research 140*, 4 (2005), 273–294.

[19] Schwaiger, H. F., Denlinger, R. P., and Mastin, L. G. Ash3d: A finite-volume, conservative numerical model for ash transport and tephra deposition. *Journal of Geophysical Research 117* (2012).

[20] Seinfeld, J. H., and Pandis, S. N. *Atmospheric Chemistry and Physics: From Air Pollution to Climate Change*, 2 ed. Wiley-Interscience, 2006.

[21] Suzuki, T. A theoretical model for dispersion of tephra. In *Arc Volcanism: Physics and Tectonics*, D. Shimozuru and I. Yokoyama, Eds. Terra Scientific Publishing Company, 1983, pp. 95–113.

[22] Webster, H., Devenish, B., Mastin, L., Thomson, D., and Van Eaton, A. Operational modelling of umbrella cloud growth in a lagrangian volcanic ash transport and dispersion model. *Atmosphere 11*, 2 (2020).

[23] Wilson, L., and Huang, T. C. The influence of shape on the atmospheric settling velocity of volcanic ash particles. *Earth and Planetary Science Letters 44*, 2 (1979), 311–324.