PUFF – Volcanic Ash Dispersion Modeling
PUFF-AFWA Version 3.00
Test Plan Document

Document Number UPOS-B33-04

L. Nguyen

25 Sept 2003

Table of Contents

Section **Page**

Table of Contents

Section **Page**

List of Figures and Tables

Name **Page**

1 Introduction

1.1 Purpose

This document is the test plan document for the PUFF volcanic ash dispersion modeling programs, which predict the geographical distribution of volcanic ash from an eruption versus time. It will be a “specification-based” software test, consistent with the PUFF Functional Requirements Document.

1.2 Background

PUFF is a volcanic ash dispersion prediction tool. PUFF was developed at the Geophysical Institute, University of Alaska Fairbanks and used by the Alaska Volcano Observatory (AVO) for volcano monitoring. Initially, PUFF was a research tool conceived by Dr. Hiroshi Tanaka for predicting the movement of eruption clouds. Dr. Craig Searcy conceived and developed the present version of PUFF as part of his PhD program. This version is used by the National Weather Service (NWS) and AVO to track volcanic eruption clouds.

Refinements in the Graphical User Interface (GUI) and data conversions were implemented by The Johns Hopkins University Applied Physics Laboratory (JHU/APL) in a joint project with the University of Alaska. JHU/APL is responsible for modifying the program and/or developing supporting utilities to facilitate its deployment at the Air Force Weather Agency (AFWA) site at Offutt AFB, NE. An additional responsibility is to develop a basic documentation set including this document. The system is currently in operation at the Air Force Weather Agency (AFWA).

1.3 Overview

The PUFF program models the dispersion of volcanic ash from an eruption and provides predictions of ash particle locations (latitude/longitude/altitude) versus time given eruption characteristics and wind field forecasts produced by another model. The PUFF application suite comprises five executable programs (puff, afwa2puff, puffgui, puffview, and ashdump) that provide the modeling capability; input data preprocessing; a graphical user interface (GUI) for model run specification; a GUI for viewing results; and utilities for viewing summaries of binary file contents. The application is written in C++, while the associated GUI functions are largely handled via the Tool Command Language (Tcl) scripts employing Toolkit (Tk) Motif widgets. The application suite can be hosted on Unix systems.

The PUFF model predicts the movement of ash particles ejected from a volcano versus time. The operator may select from a number of different initial conditions for the ash distribution and particle size. Particle locations are computed for each integration step (typically 5 minutes), with a snapshot of all particle locations at a given summation time (typically one or more hours) being written to an ash file.

For input, PUFF requires the name of a volcano, eruption characteristics, and forecasts of wind speeds for the time period of interest. The wind speed data must be available in gridded binary (GRIB) files. At AFWA, these required wind GRIB files are produced by a variety of models. PUFF's afwa2puff program converts the GRIB file outputs of the various wind models to U and V wind velocity versus geopotential height files. The U and V files serve as inputs for PUFF's volcanic ash tracking model. PUFF's puffview program displays a map of the area surrounding the volcano of interest overlaid with a graphical depiction of the ash distribution and overlaid with location identifier labels (pushpins). The PUFF volcanic ash tracking model outputs a series of ash files in Network Common Data (netCDF) format, describing the ash distribution over time.

The processes used in the model and an analysis of model results versus observations are given in “PUFF: A high-resolution volcanic ash tracking model,” (see reference 1).

1.4 Components

Table 1-1 identifies and provides a brief description of the roles of the PUFF application suite components.

Table 1-1 PUFF Suite Executables

Executable	Role
puff	Contains the volcanic ash dispersion model and is executed for each model run.
puffgui	GUI invoked by the operator and used to specify model parameters and select source wind data for use by the model; automatically invokes afwa2puff (wind data conversion), puff (model), and puffview (view results) as necessary.
puffview	GUI normally invoked automatically by puffgui following a model run. This displays a map of an area surrounding the volcano of interest overlaid with a graphical depiction of the ash distribution and overlaid with location identifier labels (pushpins).
ashdump	Utility normally invoked by puffview to extract data from the ash files produced during the model run. It can also be invoked from the command line by a knowledgeable operator to inspect ash data.
afwa2puff	Utility normally invoked by puffgui to convert wind speed data contained in gridded binary (GRIB) files into a form usable by the puff executable. Can also be invoked from the command line to automate source wind file creation.

1.5 Document Organization

Section 1 describes the scope of the PUFF application system.

Section 2 lists applicable references.

Section 3 provides an outline of the resources necessary to install, run, and evaluate the PUFF application system.

Section 4 defines the test plan and test report.

Section 5 outlines the test plan and defines its contents.

Appendix A contains the installation procedures.

Appendix B contains the requirements cross-reference matrix

Appendix C contains a list of acronyms and abbreviations.

2 References

"PUFF: A high-resolution volcanic ash tracking model," Journal of Volcanology and Geothermal Research 80 (1998) pp1-16, Craig Searcy, Ken Dean, and William Stringer

"The WMO Format for the Storage of Weather Product Information and the Exchange of Weather Product Messages in Gridded Binary Form as used by NCEP Central Operations," Clifford H. Dey, U.S. Department of Commerce, National Oceanic and Atmospheric Administration, National Weather Service, National Centers for Environmental Prediction Office Note 388, GRIB (Edition 1), 10 March 1998.

"NetCDF User's Guide for C – An Access Interface for Self-Describing, Portable Data", Version 3, Russ Rew, Glenn Davis, Steve Emmerson, and Harvey Davies, Unidata Program Center, June 1997 - http://www.unidata.ucar.edu/packages/netcdf/index.html

3 PUFF Testing Overview

3.1 Introduction

The PUFF Test Plan will be essentially a "specification-based" software test, consistent with the PUFF Functional Requirements Document. A three-part approach will be followed for the PUFF Test Plan where each part will include tests for face validity, context validity and concept validity:

Testing PUFF Installation
Testing PUFF Execution
Testing PUFF Data Output.

To facilitate the test plan, two CD's will be created: one for PUFF installation (PUFF Installation Disk) and one with the required data files (PUFF Test Data CD) for execution of the PUFF Test Case(s). Along with the PUFF User's Guide, the PUFF Test Plan should provide all the necessary code, data and information required by the tester(s). In addition, PUFF tester(s) will be provided with a set of "problem report" forms and a test log. Beyond this "specification-based" software test, these completed forms and the PUFF Test Report will enable use of the PUFF Test Plan as a "tool" to refine PUFF to ensure ease of use and successful program execution.

3.1.1 Testing PUFF Installation

In brief, following the PUFF Installation Procedures Document, the tester should be able to create a main PUFF directory that includes all relevant program codes by using the PUFF Installation CD. The directory structure and files created should be checked against the ones outlined in the Version Description document.

3.1.2 Testing PUFF Execution

Prior to testing PUFF execution the user must copy the wind model data files from the PUFF Test Data CD, i.e. the AVN, NOGAPS, MM5, AVN-datacube, and MRGF-datacube files per the User's Guide. The tester will be required to verify that all applicable files are available prior to program execution.
The test cases chosen to validate PUFF execution are intended to be representative of "typical" situations. For example, the AVD wind data with the Etna volcano in Italy is used to test many of the PUFF features that are not dependent on the data type or volcano used. Subsequent tests will specify specific data types and volcanos for testing. Successful execution of PUFF should return output consistent with the data and figures in the PUFF User's Guide.

3.1.3 Testing PUFF Data Output

PUFF data output tests are intended to verify that PUFF can save images containing the displayed map and ash distributions in postscript, gif, or animated-gif format and that the saved files do represent the puffview images and that PUFF can save ash particle data in netCDF format files where the output ash propagation is consistent with the input data (e.g. intended coverage area) and internal PUFF parameters

4 Test Plan Documentation Contents

4.1 Test Plan

The detailed PUFF Test Plan (immediately following this section) covers the entire testing effort, what is required, what is expected, and the corresponding documentation. Problem reports generated will be logged by the tester(s) and submitted to PUFF developers during PUFF testing on an as required basis. Those problems not resulting in program failure will be held until the current test phase is complete. Then they will be addressed as appropriate. The following sections will be included in the Test Plan.

Introduction (Overview): Include references to all relevant policy documents, and high level product plans; e.g. the Space Department Software Development Plan and UPOS requirements.
Test items: Software items (function, module, feature, whatever) that are to be tested. A comprehensive list or reference to a document that contains a comprehensive list. Include references to specifications (e.g., requirements and design) and manuals (e.g., user, operations, and installation).
Features to be tested: Cross-reference them to test design specifications.
Features not to be tested: Which ones and why not.
Approach: Describe the overall approach to be taken, e.g. who does it, what are the main activities, techniques and tools used for each major group of features. The criteria by which to determine when an item has been sufficiently tested.
Item pass/fail criteria: This is self-explanatory.
Suspension criteria and resumption requirements: List anything that would stop testing until fixed, what would be required to resume testing, and what tests should be redone.
Test deliverables: A list of all the documents that will be written for this product.
Testing tasks: List all tasks necessary to prepare for and do testing. Show dependencies between tasks, special skills required to do them, and when a task is required.
Environmental needs: A description of the necessary hardware, software, testing tools, facilities required to complete testing.
Roles and responsibilities: Who, what, etc.
Risks and contingencies: What high-risk assumptions are included in the test plan? What could go sufficiently wrong to significantly delay the test schedule.
Plan approvals.

4.2 Test Report

Having achieved successful PUFF execution the tester will then verify that the output data values are reasonable, as much as possible. A test report will be generated following the outline below.

Summary: What was tested (including Version ID), in what environment, and its evaluation.
Variances: Any deviation of test procedures from the specified ones and an explanation of why.
Comprehensiveness assessment: Was testing as comprehensive as the test plan called for? What modules, features, or feature combinations weren't tested enough and why?
Summary of results: What problems were reported, which were resolved and what were the resolutions? Which are still outstanding?
Evaluation: Overall evaluation of each item tested based on test results.
Summary of activities: Summarize such things as total machines used (i.e. if more than one), times such as time for program execution, and any special events or other resource uses that deserve mention.
Approvals

5 PUFF Test Plan

5.1 Introduction (Overview):

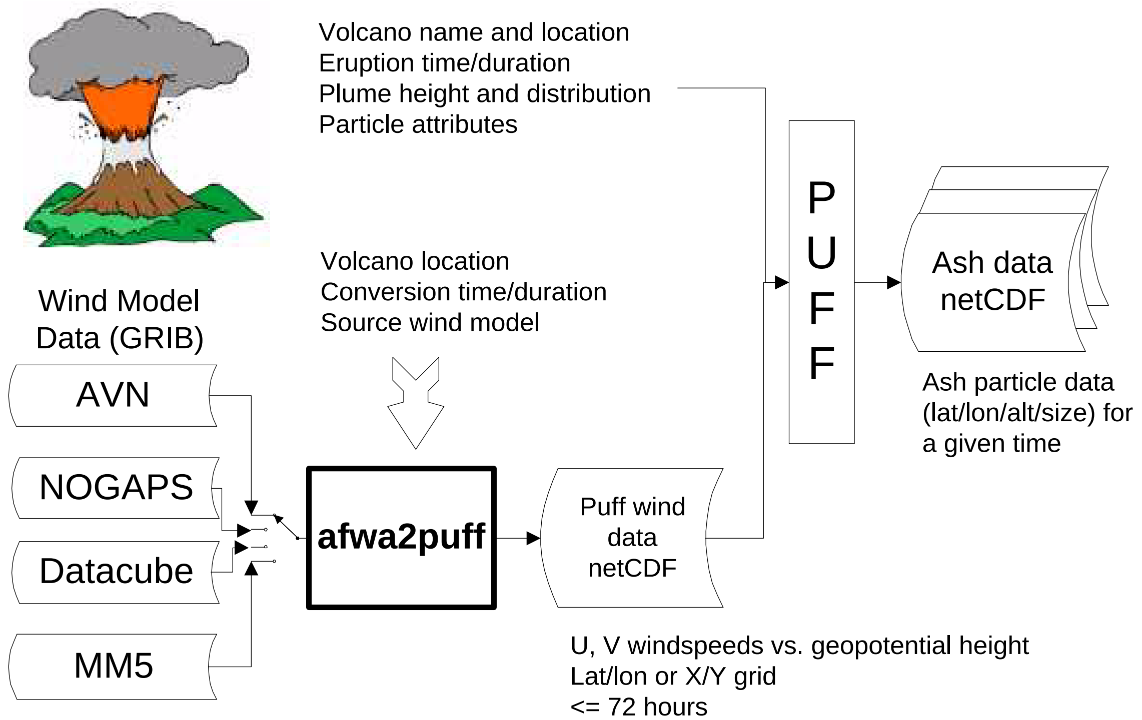
The PUFF Test Plan is intended to verify that PUFF behavior and operational characteristics (e.g. performance) are consistent with the PUFF Functional Requirements Document (FRD). As stated in the FRD, the purpose of PUFF is to use volcanic ash dispersion modeling programs to predict the geographical distribution of volcanic ash from an eruption versus time.

5.2 Test items:

The PUFF Test Plan includes testing of:

1. PUFF Installation
2. PUFF execution
3. PUFF data output

A schematic of the PUFF software architecture is shown below for PUFF Version 2.00. This schematic illustrates PUFF overall data flow and associated program drivers. A detailed data flow diagram is included in the PUFF Detailed Design Document as Figure 3-1 PUFF dataflow Diagram.



Schematic of PUFF Software Architecture

5.3 Functions and features to be tested:

5.3.1

PUFF Installation

In order to successfully execute PUFF it is critical to correctly install all components, input data files and programs, relevant to PUFF. The correctly installed PUFF directory structure is illustrated in the PUFF Version Description document Appendix A, "Inventory of Software Contents of PUFF Version 3.00". The resultant files and directories can be compared to the software inventory in the PUFF-AFWA Version Description document appendix A.

5.3.2 PUFF behavior during execution

PUFF execution requires PUFF input data including aviation (AVN), Navy Operational Global Atmospheric Prediction System (NOGAPS), Fifth Generation Mesoscale Model (MM5), AVN-datacube, and MRGF-datacube GRIB weather data files, as well as various volcano parameters. PUFF execution is them tested by observing that it does each task it is requested to do.

5.3.3 PUFF data output

PUFF data output is tested by using the output data files to see if desired results are obtained, output image files are tested by viewing them with a viewer, and output print files are tested by printing them to a color printer.

5.4 Approach:

PUFF testing will be done by members of the Johns Hopkins University/Applied Physics Laboratory Space Department SRS Group. These individuals will complete initial testing. That testing includes PUFF installation, execution and output. The specific procedure to be followed during testing is included in this PUFF Test Plan document. PUFF Tester(s) will be given a complete set of PUFF documentation:

- PUFF Functional Requirements Document
- PUFF Detailed Design Document
- PUFF User's Guide
- PUFF Test Plan

as well as any PUFF tar files, data, and/or installation CDs required.
The criteria for determining when PUFF has been sufficiently tested is:

- PUFF installation, including all required files and directory structure, is correctly accomplished via a PUFF tar file or installation CD.
- PUFF execution completely finishes and the required image output files are generated. (Any errors or exceptions that prevent PUFF execution from completely finishing are adequately documented so that the user can re-initiate PUFF execution in order to successfully finish the run.)
- PUFF output gif files… TBD.

5.5 Item pass/fail criteria:

PUFF testing will fail until the criteria above, requirements for sufficient PUFF testing, have been met.

5.6 Suspension criteria and resumption requirements:

1. All steps required for PUFF installation, execution and data output should be adequately documented, including error/exception messages, so that the user can re-initiate a PUFF run in the event of suspension. Problem report forms will be provided to testers so that if they encounter undocumented errors or exceptions they can be reported to PUFF developers. If the error or exception allows the tester to continue the PUFF run they should do so, and include a problem report for that incident with their final Test Report. If the error or exception does not allow the tester to continue the PUFF run, they should immediately report the problem to PUFF developers for correction.

5.7 Log

A PUFF Test Log has been provided to the PUFF tester(s) that includes the following headings: problem report number; date; recoverable (y/n); problem (or error) description; and solution (if recoverable). The test log is illustrated below.

Problem Report Number	***Date***	***Recoverable (y/n)***	***Problem (or error) Description***	***Solution (if recoverable***

The tester(s) will log each problem report. If an unrecoverable problem occurs it is the PUFF developers responsibility to address this problem in a timely fashion so that testing can continue. If a "recoverable" problem occurs it will be addressed, as required, by PUFF developers.

5.8 Test deliverables:

Specific PUFF test deliverables include problem reports for any unresolved problems with PUFF installation, undocumented errors or exceptions during PUFF execution or output. In addition, a final PUFF Test Report is due within one week of the completion of PUFF testing.

5.9 Testing tasks:

PUFF tester(s) should be sufficiently familiar with Unix workstation and PC operation that they are able to perform the functions necessary for PUFF installation, verification of necessary files and directory structure, program execution, simple operations or troubleshooting per PUFF displayed messages, and displaying and verification of PUFF output. Specific tasks are detailed in the PUFF Test Procedure document.

5.10 Environmental needs:

Access to a Unix Workstation, comparable to those described in the PUFF Development Environment, Section 3.1.2 of the PUFF Functional Requirements Document, is required. Sufficient disk space must be available for the required PUFF directories, files and output.

5.11 Roles and responsibilities:

PUFF testing will be done by members of the Johns Hopkins University/Applied Physics Laboratory Space Department SIS Group. The role, and responsibility, of these testers will be to provide an independent test of PUFF installation, execution and data output. Validation of PUFF output data is by the testers. The role of the PUFF developers will be to support a successful test effort. It is the responsibility of the PUFF developers to be responsive to problems, should they arise, during testing, in a timely manner. In addition, the PUFF developers will be responsible for answering questions from the tester(s) regarding input parameters during the tester(s) orientation period (as scheduled).

5.12 Risks and contingencies:

The PUFF Test Plan, as outlined, is low-risk. Only the unlikely event of a “fatal” error, from which the tester(s) could not recover and the developers could not address, would result in a significant delay to this schedule.

6 Test Procedures

The following sections discuss the test procedure specification for each phase of PUFF testing; installation, execution, and data output. In addition to PUFF documentation and related materials, tester(s) have been provided with problem report forms and a PUFF Test Log. When any problem occurs, the tester is required to complete, as appropriate, a problem report form and enter it into the PUFF Test Log. These completed forms and the PUFF Test Log will be attached as an appendix to the PUFF Test Report.

6.1 PUFF Installation Test Procedure

The following sections apply to tests on PUFF installation procedures, documentation and related materials (installation CDs).

6.1.1 Installation Test

6.1.1.1 Purpose

The purpose of this test is to verify that PUFF installation is correctly completed following the directions given in the PUFF-AFWA Installation Procedure and using the related materials distributed with the respective versions of PUFF. This verification includes all directories and files required to successfully run PUFF.

6.1.1.2 Procedure steps

§ Proceed

Follow the installation steps found in the PUFF-AFWA Installation Procedures document (a copy of which is included in Appendix A).

§ Verify

Verify that all the PUFF directories and files have been installed consistent with the PUFF Version Description document Appendix A.

6.2 PUFF Execution Test Procedures

The following sections apply to tests on PUFF execution, including data input. These tests require completion of the PUFF installation test and access to a Unix workstation, comparable to those described in Section 3.1.2 of the PUFF Functional Requirements Document. The canned wind model data from the PUFF Test Data is to be used for testing and its eruption time and duration must be known by the tester. The path to this data must be set in the puff.args file and PUFF compiled after it was set. Appendix B contains a Requirements Cross-Reference Matrix, which matches requirements to tests.

6.2.1 Basic PUFF Test

6.2.1.1 Purpose

The purpose of this test is to verify that the basic functions of PUFF work, which will confirm that PUFF was compiled and set up correctly. This test selects a volcano, generates wind data, displays a map with ash dispersion.

6.2.1.2 Procedure steps

§ Proceed

Run the PUFF application using SHISHALDIN as the volcano, set the eruption time, run the wind conversion (file | grib convert | create puff files), and finally run puffview (file | run).

§ Verify

Verify that PUFF ran, that the wind conversion ran and that the map came up showing the Shishaldin volcanic ash map.

6.2.2 PUFF with no pushpins selected Test

6.2.2.1 Purpose

The purpose of this test is to show that without pushpins on the map, the map display is the same as with previous versions of PUFF. This test selects a volcano, displays a map with ash dispersion, and turns off pushpins.

6.2.2.2 Procedure steps

§ Proceed

Run the PUFF application using ETNA as the volcano, set the eruption time, run the wind conversion, run puffview, and finally turn off all pushpin types (click country, other, and state).

§ Verify

Verify that the display shows the Etna volcanic ash map with no pushpins.

6.2.3 PUFF with all but a few pushpins selected Test

6.2.3.1 Purpose

The purpose of this test is to verify that the user can select all pushpins and also deselect a few chosen pushpins from being displayed on the map display. This test selects a volcano, displays a map with ash dispersion, turns on all pushpins, and turns a few chosen pushpins off.

6.2.3.2 Procedure steps

§ Proceed

Run the PUFF application using ETNA as the volcano, set the eruption time, run puffview, turn on all pushpin types (click AFB, airport, capital, and city), turn off London pushpin (tools | pushpins selection | London | Accept Selections), and finally turn the London pushpin on and the Heathrow pushpin off.

§ Verify

Verify that the display shows the Etna volcanic ash map with many pushpins, verify it then shows the map without the London pushpin, and verify it then shows the map with the London pushpin but not the Heathrow pushpin.

6.2.4 NOGAPS Data Test

6.2.4.1 Purpose

The purpose of this test is to verify PUFF can generate wind data using the new format of NOGAPS wind model data. This test selects a volcano, selects NOGAPS data, generates wind data, and displays a map with ash dispersion.

6.2.4.2 Procedure steps

§ Proceed

Run the PUFF application using SHISHALDIN as the volcano, set the eruption time, set wind model to nogaps, run the wind conversion and run puffview.

§ Verify

Verify that PUFF ran, that the wind conversion ran and that the map came up showing the Shishaldin volcanic ash map.

6.2.5 Run Multiple Eruption Test

6.2.5.1 Purpose

The purpose of this test is to verify PUFF can run a multiple eruption. The test generates an initial eruption then generates a second eruption 18 hours later. The test can alternately be run with two different volcanos as long as they are sufficiently close to be covered by the same wind file. It may be easier to verify that two eruptions occurred if different volcanos are used.

6.2.5.2 Procedure steps

§ Proceed

Run the PUFF application using SHISHALDIN as the volcano. Set the eruption time to 00:00 hours, the Simulation Length to 24 hours, the Save Interval to 6 hours, and the Eruption Duration to 6 hours. Run the wind conversion and select “File -> Run ...”. Wait for Puffview to come up. Go back to the main Puff window and set the eruption time to 18:00, maintaining the same date as the first eruption. Set the Simulation Length to 6 hours, the Save Interval to 3 hours, and the Eruption Duration to 6 hours. Select “File -> Run Multiple Eruption ...”. When prompted, open the ash file from the previous puff run with time stamp 1800. The file should be of the format *.1800_ash.cdf.

§ Verify

Verify that after the initial run, Puffview comes up showing an eruption at Shishaldin. Verify that one of the ash files is at 18:00 (18hr). (Note that if some ash were out of bounds, the latest ash file may have an earlier time stamp. Repeat the test using a wind conversion that has increased range.) Verify that after the second run (the multiple eruption), Puffview comes up showing dispersed ash from the first eruption in addition to a new eruption at Shishaldin.

6.2.6 Run With Existing Cloud Test

6.2.6.1 Purpose

The purpose of this test is to verify PUFF can generate a cloud file and run a simulation using the cloud file. This test creates an existing ash cloud near the volcano Shishaldin, and displays a map simulating the movement of that ash cloud.

6.2.6.2 Procedure steps

§ Proceed

Run the PUFF application using SHISHALDIN as the volcano, set the eruption time, and open the Cloud Editor. Create a cloud near the volcano which has 3 vertices and one interior elevation point of 18000 meters. Save the cloud as shishaldin.cld. Close the cloud editor. Run the wind conversion and select “File -> Run w/ Existing Cloud ...”. When prompted, open the cloud file “shishaldin.cld”.

§ Verify

Verify that PUFF ran, that the wind conversion ran and that the map came up showing the Shishaldin volcanic ash map.

6.2.7 Run New Eruption With Existing Cloud Test

6.2.7.1 Purpose

The purpose of this test is to verify PUFF can run a multiple eruption where the previous eruption is represented by an existing ash cloud.

6.2.7.2 Procedure steps

§ Proceed

Run the PUFF application using SHISHALDIN as the volcano, set the eruption time, and run the wind conversion. Open the Cloud Editor and create a cloud near Shishaldin but not directly over it. Select "File -> Run New Eruption w/ Existing Cloud ...". When prompted, open the cloud file "shishaldin.cld".

§ Verify

Verify that PUFF ran, that the wind conversion ran and that the map came up showing the Shishaldin volcanic ash map. Verify that there is a new eruption at Shishaldin as well as dispersed ash from the existing ash cloud, shishaldin.cld.

6.2.8 Input GRIB Format Data Test

6.2.8.1 Purpose

The purpose of this test is to verify PUFF can obtain input data from records in GRIB format files that are based on various grid types. This test requires the existence of GRIB files in the tester's file system containing data in each of the following grid types:

Latitude/Longitude Grid
Polar Stereographic
Lambert Conformal
Mercator

6.2.8.2 Procedure steps

§ Proceed

Use wgrib-afwa to confirm the grid type used in your GRIB format input files by issuing the following command at the Unix prompt,

wgrib-afwa -V -d 0 <your GRIB file>

where <your GRIB file> is the GRIB format wind model file that you are using as input to PUFF. The fifth line of output describes the grid type. Confirm that it is one of the types listed above.

Run the PUFF application, select a volcano, set the eruption time, and set the wind model to match the wind model file used to confirm the grid type. Run the wind conversion and run puffview.

§ Verify

Verify that PUFF ran, that the wind conversion ran and that the map came up showing the volcanic ash map.

6.2.9 Generate Images and Animations Test

6.2.9.1 Purpose

The purpose of this test is to verify PUFF can generate image and animated GIF files. This test selects a volcano, generates wind data, displays a map with ash dispersion, and saves the map.

6.2.9.2 Procedure steps

§ Proceed

Run the PUFF application using SHISHALDIN as the volcano, set the eruption time, run the wind conversion (file | grib convert | create puff files), and run puffview (file | run). Select file | save from the main puffview menu to open the Ash Save dialog, then highlight the first ash item in the "Select Ash to Save:" list on the left. Select "Single gif" from the "Save as:" drop-down menu, set the Filename to single_gif.gif, and press "Save". Re-open the Ash Save dialog and highlight all of the ash items in the "Select Ash to Save" list. Select "Animated gif" from the "Save as:" drop-down menu, set the Filename to animated_gif.gif, and press "Save". Do not close puffview.

§ Verify

Verify that PUFF wrote the files single_gif.gif and animated_gif.gif to the current working directory. Verify that single_gif.gif matches the first ash map in puffview, and that when opened in a movie player or web browser, animated_gif.gif displays the same ash animation as seen in puffview when the Loop button is pressed.

Verify that the gif images display the ash height bar legend, the volcano name, latitude, and longitude, the eruption date and time, the forecast time for this ash distribution, and the date and time the image was created.

6.3 PUFF Data Output Test Procedure

The following sections apply to tests on PUFF output data.

6.3.1 Data Output Check Test

6.3.1.1 Purpose

The purpose of this test is to verify that the latest PUFF application delivers the identical data output that the previous version of PUFF did. This test selects a volcano, generates wind data, displays a map with ash dispersion, thus generating netCDF ash data files. Then, operating from a different directory, the same wind data is input to the previous version of PUFF again generating the netCDF files. The two outputs are then compared.

6.3.1.2 Procedure steps

§ Proceed

Run the PUFF application with the seed parameter set to 100 (puffgui –seed "100"), set SHISHALDIN as the volcano, set the eruption time, run the wind conversion, and run puffview which outputs the netCDF files. In another test directory, run the previous version of PUFF with the seed parameter set to 100, set SHISHALDIN as the volcano, set the same eruption time, run the wind conversion, and run puffview which again outputs the netCDF files. Use the diff command to compare the output netCDF files.

§ Verify

Verify that PUFF ran, that the wind conversion ran, that the map came up showing the Shishaldin volcanic ash map, and that the netCDF files were written. Verify the same steps were taken using the previous version of PUFF. Finally verify that the files generated were identical.

6.4 Wrap up

Remove the PUFF main directory and all its contents from the test platform. The PUFF-AFWA Installation Procedures document has removal instructions.

7 APPENDIXES

APPENDIX A - Installation Procedures

The UNIX installation CD contains the tar file. Listed below are the instructions to install the software on a UNIX host machine:

- Load the CD labeled "*PUFF-AFWA Software Version 3.00*" into the CD drive

- If your CD drive is on a PC, then FTP the tar file from the CD drive to the UNIX host machine, to the directory **<your installation directory>**.

- If your CD drive is on the UNIX host machine, then copy the tar files from the CD drive to the directory **<your installation directory>**.

- Log on to the host machine and change directory to the top level PUFF-AFWA directory:
  cd <your installation directory>.

- Make a new version directory on the host system with the following command:
  mkdir <your installation directory>.

- Extract the software files by issuing the following commands:
  tar –xvf puff-afwa-v300.tar

- Review and edit the .args files, which contain settings and pointers which are dependent on the software version, the user's file structure and the location of wind data files.

  puff.args - Contains settings, paths and wind file directory defaults.
  puffview.args - Contains display and path defaults.
  ashdump.args - Contains ash output defaults.
  uni2puff.args - Contains unidata defaults.

  - (In viewgmt directory)

    cd <your installation directory>/src/viewgmt

    more puffview.args

    (if necessary) vi puffview.args

  - (In puffsrc directory)

    cd <your installation directory>/src/puffsrc

    more puff.args

    (if necessary) vi puff.args

    more ashdump.args

    (if necessary) vi ashdump.args

    more uni2puff.args

    (if necessary) vi uni2puff.args

The following table lists the arguments that must be edited and the file(s) in which they reside. If data is not available for all models, it is sufficient to specify only the path of the GRIB files whose model type matches the "model" argument.

ARG Name	Value	File(s)
avnDatacubePath	<path to your AVN-based Datacube GRIB files>	puff.args
avnPath	<path to your AVN model GRIB files	puff.args
mergeDatacubePath	<path to your AVN and merged MM5-based Datacu GRIB files>	puff.args
mm5Path	<path to your MM5 model GRIB files	puff.args
nogapsPath	<path to your NOGAPS model GRIB files>	puff.args
model	<wind model name (avn, nogaps, mm5, avn-datacube, or merge-datacube)>	puff.args
pinPath	<your installation path>/src/viewgmt/pushpins.txt	puff.args, puffview.arg
pushpinCfg	<your installation path>/src/viewgmt/pushpins.cfg	puff.args, puffview.args
volcPath	<your installation path>/src/puffsrc/volcanos.	puff.args, puffview.args

- Establish a series of symbolic links. These are not reproduced when tar'ing and they need to be done exactly as described.
  - (In puffsrc directory)

    cd <your installation directory>/src/puffsrc

    ```
    ln –s args.C ashdump_args.C
    ln –s args.C puff_args.C
    ln –s args.C uni2puff_args.C
    ```

  - (In dstsrc directory)

    cd <your installation directory>/src/dstsrc

    ```
    ln –s ../puffsrc/puff.args afwa2puff.args
    ln –s ../puffsrc/args.C afwa2puff_args.C
    ```

- (In gui directory)

  cd <your installation directory>/src/gui

  ```
  ln –s ../puffsrc/puff.args puffgui.args
  ln –s ../puffsrc/args.C puffgui_args.C
  ```

- (In viewgmt directory)

  cd <your installation directory>/src/viewgmt

  ```
  ln –s ../puffsrc/args.C puffview_args.C
  ```

- Verify the following environment variables are set and correct using the 'env' command. They can be set in your .cshrc file so that they will always be active. Below are sample settings, adjust them for your configuration as needed:

  setenv OPENWINHOME <your winhome directory>

  setenv GMTHOME <your software installation path>/<your hosttype>/GMT3.3.4

  set path= (<your installation directory>/src/bin <your software installation path>/<your hosttype>/bin $OPENWINHOME/bin <your SUNWspro path>/bin $path)

- Modify the Makefile to point to your installation paths:

  cd <your installation path>/src

  (if neccessary) vi Makefile

- Compile all the software by issuing the following commands, using a gcc 3.0 or higher version compiler:

  cd <your installation directory>/libsrc

  make clean

  make all

  cd <your installation directory>/src/

  make clean

  make all

APPENDIX B – Requirements Cross-Reference Matrix

Verification methods are (A) analysis, (I) inspection, (D) demonstration, and (T) test.

Requirement Paragraph	Test Procedure Paragraph	Verification Method
FUNCTIONAL REQUIREMENTS		
The program shall provide the capability to perform the following functions:		
Predict ash distributions versus time given wind data for the area of interest and characteristics of the volcanic eruption.	6.2.1	D,T
Model Invocation		
The application shall provide the operator the capability to invoke the modeling action via the following methods:		
(a) GUI	6.2.1	D
(b) Command line	6.2.1	D
Modeling Options		
The application shall provide the operator the capability to specify the following eruption-related characteristics:		
(a) Volcano name	6.2.1	D
(b) Volcano regional location	6.2.1	D
(c) Volcano latitude and longitude	6.2.1	D
(d) Volcano cone elevation	6.2.1	D
(f) Eruption date and time	6.2.1	D
(g) Eruption duration	6.2.1	D
(h) Minimum plume height (linear distribution only)	6.2.1	D
(i) Maximum plume height	6.2.1	D
(j) Vertical distribution of particles in plume	6.2.1	D
(1) Linear	6.2.1	D
(2) Poisson	6.2.1	D
(3) Exponential	6.2.1	D
(k) Maximum horizontal extent of particles in plume at plume peak (defines an inverted cone with 0 horizontal extent at plume base)	6.2.1	D

Requirement Paragraph	Test Procedure Paragraph	Verification Method
(l) Nominal vertical extent of particles below plume peak for Poisson and Exponential distributions	6.2.1	D
(m) Initial particle count (number of ash particles)	6.2.1	D
(n) Mean of a Gaussian distribution of particle sizes (in meters) on a logarithmic scale	6.2.1	D
(o) Standard deviation of a Gaussian distribution of particle sizes on a logarithmic scale	6.2.1	D
(p) Restart file name for multiple eruptions and existing ash clouds	6.2.1	D
The application shall provide the operator the capability to specify the following simulation characteristics:		
(a) Simulation run time	6.2.1	D
(b) Time intervals at which ash locations are saved to files	6.2.1	D
(c) Integration time (time interval at which ash locations are computed)	6.2.1	D
(d) Windfield data source	6.2.1	D
Modeling and Simulation Parameter Input Options		
The application shall provide the capability to specify modeling options via the following methods:		
(a) GUI	6.2.1	D
(b) Text file	6.2.1	D
(c) Command line arguments	6.2.1	D
Windfield Data Input		
The application shall be capable of obtaining U wind speed, V wind speed, and geopotential height versus pressure level data from records in GRIB format files based on the following grid types:		
(a) Latitude/longitude	6.2.8	D,T
(b) Cartesian	6.2.8	D,T
For Cartesian gridded data, the program shall accept data described in terms of the following map projections:		
(a) Polar Stereo	6.2.8	D,T
(b) Lambert Conformal	6.2.8	D,T
(c) Mercator	6.2.8	D,T

Requirement Paragraph	Test Procedure Paragraph	Verification Method
Ash Location Data Output		
The application shall record the location of ash particles at simulation time intervals specified by the save file interval.	6.2.1	T,D
Display the ash distribution predictions overlaid on a map, using color to differentiate particles at different altitudes.	6.2.1	T,D
View Options		
The application shall provide the operator the capability to specify the following view options:		
Map projections		
(a) North Polar Stereo for areas entirely in the Northern Hemisphere	6.2.1	D,I
(b) South Polar Stereo for areas entirely in the Southern Hemisphere	6.2.1	D,I
(c) Mercator for areas between 85 deg North latitude and -85 degrees South latitude	6.2.1	D,I
Range of map display by specifying latitude/longitude points of the desired extent	6.2.1	D,I
Selection of one or more ash files to open	6.2.1	D,T
Selection of one or more open ash files to display	6.2.1	D,T
Animation of ash motion via sequential display of data from successive ash files	6.2.1	D,T
Association between ash symbol colors and ash elevation		
(a) Choice of 16 predefined colors for any defined level	6.2.1	D
(b) Mapping of ash altitude to one of at least one, but no more than 16, distinct levels	6.2.1	D
(c) Altitude of highest ash displayed	6.2.1	D
(d) Altitude of lowest ash displayed	6.2.1	D
Selection of ash symbol size in range from no symbol (individual point) to 10 pixel symbols	6.2.1	D
Produce a Postscript file containing the map depiction and ash overlay and send it to a user-specified print queue or save the file to a user-specified filename.	6.2.1	D,T
Selection of ash symbol color tables.	6.2.1	D

Requirement Paragraph	Test Procedure Paragraph	Verification Method
INPUT DATA		
Constraints on input data:		
(a) Eruption duration: specified in hours; must be <= simulation duration.	6.2.1	T,I
(b) Minimum plume height: 0 <= minimum plume height < maximum plume height (integer – feet)	6.2.1	T,I
(c) Maximum plume height: minimum plume height < maximum plume height <= 66,000 (integer – feet)	6.2.1	T,I
(d) Initial particle count (number of ash particles): 0 < count (integer)	6.2.1	T,I
(e) Simulation run time: 1 < time <= 72 (real – hours)	6.2.1	T,I
(f) Time interval at which ash files are saved: 1 <= interval <= simulation duration (integer – hours)	6.2.1	T,I
(g) Integration interval: 1 <= interval <= 60 (integer – minutes)	6.2.1	T,I
Selection of source windfield data:		
The program shall provide the operator with the capability to convert U wind speed, V wind speed, and geopotential height versus pressure level records from GRIB files into U wind speed and V wind speed versus geopotential height netCDF files with the following specifiable options:	6.2.1, 6.3.1	D,T
Source wind model from the following choices: aviation (AVN), Navy Operational Global Atmospheric Prediction System (NOGAPS), Fifth Generation Mesoscale Model (MM5), AVN-datacube, and MRGF-datacube	6.2.1	D,T
Eruption time (earliest time for which wind data are required).	6.2.1	D,T
Total time coverage of interest. This can range from 1 hours to 72 hours based on the assumed availability of 72 hours of forecast data.	6.2.1	D,T

Requirement Paragraph	Test Procedure Paragraph	Verification Method
Minimum and desired numbers of hours of data to convert. The program shall consider it a success to find at least the minimum number of forecast hours but shall seek to satisfy the desired number if available. If at least the minimum number of hours are not yet available for a given model run, the function shall examine the next previous model run, until all available model runs are exhausted.	6.2.1	D,T
Relative latitude and longitude range (a single value for each) around the volcano (presumed to be the center point of the windfield data to be converted) for the case of global data sources.	6.2.1	D,T
Theater and nest of interest for the MM5 model data source	6.2.1	D,T
Path to the netCDF output file destination directory	6.2.1	D,T
The program shall distinguish GRIB source files for the various wind models by directory. For all cases except MM5, the files are contained in the "parent" directory for that model. For MM5, there are subdirectories in the parent directory for each defined theater. The files are stored in the theater directory corresponding to their geographical coverage.	6.2.1	D,T
The program shall maintain an association between default directory and source wind model and provide the capability for the user to override these defaults on the command line or via a text input file upon program invocation.	6.2.1	D,T
The program shall use the wind model output from the latest wind model cycle that covers the time period from the operator-specified eruption time to the operator-specified minimum number of hours hence.	6.2.1	D,T
The program shall support specification of forecast filename conventions for each source model, providing a simple method for specifying and changing them. These conventions can change, and such changes should not necessitate source code modifications.	6.2.1	D,T
In the GUI case, the program shall provide the capability to select the input windfield files for use by specifying the model and selecting a cycle (the time of the initial forecast) from a list of those available. For the MM5 case, the program shall list the theater and nest for each cycle.	6.2.1	D,T

Requirement Paragraph	Test Procedure Paragraph	Verification Method
In the GUI case, the program shall display information about the latitude/longitude/time coverage of a selected input windfield file.	6.2.1	D,T
PROCESSING		
For each particle in the ash set the program shall calculate the total distance and direction of movement for each integration interval.	6.2.1	D,I
The program shall derive vertical wind velocities for each grid point present in the U and V windfield data.	6.2.1	D,I
The program shall apply the following processes (as described in Reference 1) to determine ash particle movement:		
Transport – based on wind velocity at particle location		A
Turbulent dispersion – based on a random walk process		A
Fallout – based on particle size		A
The program shall terminate processing following completion of the specified analysis time or at the completion of the first integration step where a particle is determined to have left the area covered by the windfield data, whichever comes first.	6.2.1	D,T
OUTPUT DATA		
The program shall save ash particle location data in a netCDF file format with longitude, latitude, geopotential height, size, and age data for each particle.	6.3.1	T,I
The program shall provide as an option the capability to save the derived vertical wind velocity file created during data preprocessing.	6.3.1	T,I
The program shall provide as an option the capability to save the initial ash data (ash distribution at start of simulation).	6.3.1	T,I
The program shall maintain a log during the model execution that contains summary information about the extent of the input windfield files and the values of modeling and simulation options in use for the run.	6.3.1	T,I

Requirement Paragraph	Test Procedure Paragraph	Verification Method
The program shall provide the capability to save an image containing the displayed map and ash distribution(s) as follows:		
The program shall provide the following options for saved file format:		
(a) Postscript	6.2.9	D,T
(b) Graphics Interchange Format (GIF)	6.2.9	D,T
(c) Animated GIF (if two or more ash files are available)	6.2.9	D,T
The program shall include the following ancillary information in the image:		
(a) Ash height bar legend	6.2.9	D
(b) Volcano name	6.2.9	D
(c) Volcano latitude and longitude	6.2.9	D
(d) Eruption date and time	6.2.9	D
(e) Date and time image was created		
(f) Date and time of forecast for which this ash distribution is valid	6.2.9	D
The program shall provide default values in pixels for the width and height of GIF and animated GIF files and shall provide the capability for the operator to override the defaults on a case-by-case basis.	6.2.9	D
The program shall provide the capability to save a file describing the geometry and ash density of an existing ash cloud.	6.2.9	D
AFWA2PUFF REQUIREMENTS		
The afwa2puff conversion program shall provide the capability to perform the following functions:		
Produce PUFF input files conforming to the format, organization, and content specified in section 5.2.2 (of Puff Requirements Document) from GRIB sources containing U and V wind speed data and geopotential height data versus pressure level.		I,T
Allow operator selection of the source wind model from the following choices:		

Requirement Paragraph	Test Procedure Paragraph	Verification Method
(a) aviation (AVN)	6.2.1	D,T
(b) Navy Operational Global Atmospheric Prediction System (NOGAPS)	6.2.1	D,T
(c) Fifth Generation Mesoscale Model (MM5)	6.2.1	D,T
(d) AVN-datacube	6.2.1	D,T
(e) MRGF-datacube	6.2.1	D,T
Allow operator specification of the forecast coverage time extent in hours up to and including 72 hours.	6.2.1	D,T
For global latitude/longitude based GRIB files, allow operator selectable latitude and longitude ranges bounding the point of interest for which wind speed data will be converted.	6.2.1	D,T
For MM5 data source, allow operator selection of theater and nest of interest.	6.2.1	D,T
Be operable from the command line or via a graphical user interface (GUI) integrated with the PUFF application front end GUI (puffgui).	6.2.1	D,T
For the case where MM5 data will be converted via the GUI interface, dynamically determine which theaters have been defined, and following operator selection of a theater from this set, dynamically determine and present to the operator nests available for selection.	6.2.1	D,T
Support specification of forecast filename conventions for each source model, providing a simple method for specifying and changing them. These conventions can change, and such changes should not necessitate source code modifications. Following are current descriptions:	6.2.1	D,T
AVN: Zero hour forecast/analysis data in "gblav.TxxZ.PGrbF00", where xx represents the cycle hours and can be 00 or 12 at this time. They may eventually add 06 and 18 model runs. Subsequent forecast files are gblav.TxxZ.PGrbF03 – F78 and may not necessarily step in three-hour increments.	6.2.1	D,T

Requirement Paragraph	Test Procedure Paragraph	Verification Method
NOGAPS: Zero hour forecast/analysis data in “nogaps.TxxZ.anl”, where xx represents the cycle hours and can be 00 or 12 at this time. They may eventually add 06 and 18 model runs. The next forecast file in time sequence is nogaps.TxxZ.000 (1 hour forecast). Subsequent forecast files are nogaps.TxxZ.003 – 096, and may not necessarily step in three-hour increments.	6.2.4	D,T
Datacube: (applies to both AVN and MRGF) Zero hour forecast/analysis data in YYYYDOYHHMM0000, where YYYY is the 4-digit year, DOY is the three digit Julian date, HHMM are the hours and minutes of the cycle time, and 0000 represent the forecast hours. Subsequent forecasts are YYYYDOYHHMM0000 – 0144.	6.2.1	D,T
MM5: Zero hour forecast in us057g1010txxyHHMMhhhmm, where xx represents the theater number, y represents the nest (only the “a” nest is guaranteed to have a 0 hour forecast), HHMM are the hours and minutes of the base cycle and can be anything, and hhhmm are the hours and minutes of the forecast, and will be all zeros for the base file in the series. Subsequent forecast files will differ from the base in that the nest will reflect the nest desired by the PUFF operator and the hhhmm values will reflect the forecast time. These times do not have to be on even hours, and the interval between forecast files currently varies from one to three hours, but can be anything.	6.2.1	D,T
Alert the operator if it detects a wind model output file otherwise meeting usability criteria but having file permissions that prohibit its use by the program process. In such a case, the program shall continue looking for additional wind data for that run.	6.2.1	D,T
Convert wind speed data for all grid points in MM5 files.	6.2.1	D,T
Convert wind speed data for a user-specifiable subset of grid points centered on the volcano location for global latitude/longitude grids. The program shall provide the capability for the user to specify a symmetric range in latitude and longitude around the volcano location. If the delta latitude specification and volcano location are such that the latitude coverage extends over either pole, global longitude data must be converted for the requested range of latitudes.	6.2.1	D,T

Requirement Paragraph	Test Procedure Paragraph	Verification Method
Provide the capability for the user to specify minimum and desired numbers of hours of data to convert. The program shall consider it a success to find at least the minimum number of forecast hours but shall seek to satisfy the desired number if available. If at least the minimum number of hours are not yet available for a given model run, the function shall examine the next previous model run, until all available model runs are exhausted.	6.2.1	D,T
Provide a progress indication approximating the percentage complete (in terms of overall processing time).	6.2.1	D
Accept operator-specified values via command line arguments and via a text input file whose filename is provided as a command line argument.	6.2.1	D,T
INPUT DATA		
Input parameter specification		
Methods supported		
The program shall support specification of input parameters as argument-value pairs on the command line, and as argument-value pairs given in an ASCII file whose filename is provided in an argument-value pair on the command line.	6.2.1	I
Parameters		
(a) Volcano latitude: Real number [-90.0, 90.0] degrees North	6.2.1	I,T
(b) Volcano longitude: Real number [0.0, 360.0) degrees East	6.2.1	I,T
(c) Maximum latitude delta: Real number [15.0, 180.0] degrees	6.2.1	I,T
(d) Minimum latitude delta: Real number [-15.0, -180.0] degrees	6.2.1	I,T
(e) Maximum longitude delta: Real number [30.0, 180.0] degrees	6.2.1	I,T
(f) Minimum longitude delta: Real number [-30.0, -180.0] degrees	6.2.1	I,T
(g) Minimum number of forecast hours to convert: Integer number [1, 72]	6.2.1	I,T
(h) Desired number of forecast hours to convert: Integer number [1, 72]	6.2.1	I,T
(i) Source wind model	6.2.1	I,T

Requirement Paragraph	Test Procedure Paragraph	Verification Method
(j) Paths to parent directories for each wind model: Max of 180 characters each.	6.2.1	I,T
(k) Directory in which to store resulting PUFF input files	6.2.1	I,T
(l) Eruption time (UTC string) – Format "YYYY MM DD:HH"	6.2.1	I,T
(m) Theater (MM5 model only)	6.2.1	I,T
(n) Nest (MM5 model only)	6.2.1	I,T
Default values		
The program shall support definition of default values for the following parameters:		
(a) Maximum latitude delta	6.2.1	I
(b) Minimum latitude delta	6.2.1	I
(c) Maximum longitude delta	6.2.1	I
(d) Minimum longitude delta	6.2.1	I
(e) Minimum number of forecast hours to convert	6.2.1	I
(f) Desired number of forecast hours to convert	6.2.1	I
(g) Source wind model	6.2.1	I
(h) Paths to parent directories for each wind model	6.2.1	I
(i) Directory in which to store resulting PUFF input files	6.2.1	I
Source wind file selection		
Distinguish GRIB source files for the various wind models by directory as follows:		
For all cases except MM5, the files are contained in the "parent" directory for that model.	6.2.1	I
For MM5, there are subdirectories in the parent directory for each defined theater. The files are stored in the theater directory corresponding to their geographical coverage.	6.2.1	I
Maintain an association between default directory and source wind model and provide the capability for the user to override these defaults on the command line or via a text input file upon program invocation.	6.2.1	I,D
Use the wind model output from the latest wind model cycle that covers the time period from the operator-specified eruption time to the operator-specified minimum number of hours hence.	6.2.1	D,T

Requirement Paragraph	Test Procedure Paragraph	Verification Method
Be capable of determining the applicable model cycle and the range of valid forecast times for all nests. There will always be an “a” nest file series for a given MM5 theater. The “a” nest always includes a zero hour forecast file (meaning applicable at the cycle time for the model run). The other nests (e.g., b, c, d) do not have to start, and usually do not start, with the model cycle time.	6.2.1	D,T
Select candidate source files within a directory as follows:		
(a) filenames match filename convention;	6.2.1	I,T
(b) files were produced by a single model run whose cycle started at or before operator-specified eruption time;	6.2.1	I,T
(c) sufficient forecast hour coverage exists to cover from the eruption time to eruption time plus and operator-specified minimum number of hours	6.2.1	D,T
Determine the order of storage of data in a GRIB file from the information in the GRIB Grid Description Section (GDS) header.		A
PROCESSING		
The program shall be capable of performing the following processing steps:		
Data selection		
Sufficient source data determination		
The program shall be capable of determining the range of source wind file data needed to satisfy a user’s request given the user’s specification of the minimum and desired number of hours to convert and the volcano eruption time.	6.2.1	D,T
The program shall notify the user if data for the minimum time coverage is not available.	6.2.1	D,T
Grid subset extraction		
The program shall provide the capability to select a subset of the source wind file data points for cases where the source wind model data is projected on a global latitude/longitude grid as described in the following requirements:	6.2.1	D,T
Delta latitude specification		
The program shall provide the capability to select only those source wind file data points corresponding to a latitude range	6.2.1	D,T

Requirement Paragraph	Test Procedure Paragraph	Verification Method
defined by the volcano’s latitude and the positive and negative delta latitude specified by the user.		
In case either the Northerly or Southerly latitude extent was to extend over the pole, the program shall limit latitude extent to the pole and shall extend longitude coverage to complete coverage (i.e., 0-360 degrees longitude coverage).	6.2.1	D,T
Delta longitude specification		
The program shall provide the capability to select only those source wind file data points corresponding to a longitude range defined by the volcano’s longitude and the positive and negative delta longitude specified by the user.	6.2.1	D,T
Data ordering		
The program shall be capable of determining from the GRIB header the ordering of data stored in a wind source file (any of four possible orders: increasing X, increasing Y; increasing X, decreasing Y; decreasing X, increasing Y; decreasing X, decreasing Y) and reordering it to the increasing X, increasing Y order required by PUFF.		A
Data conversion		
The program shall be capable of converting wind speed data versus pressure level (in hPa) as provided by the GRIB files to wind speed data versus geopotential height (in meters) needed by PUFF.		A
OUTPUT DATA		
The program shall produce two netCDF files containing wind speed values on a four-dimensional grid (forecast time, geopotential height, longitude/X, latitude/Y), one for U wind speed and one for V wind speed, as described in Section 5.2.2.	6.2.1, 6,3,1	D,T

APPENDIX C – Acronyms and Abbreviations

AACGM	Attitude Adjusted Corrected Geomagnetic
ACE	Advanced Composition Explorer
AFCCC	Air Force Combat Climatology Center
AFOSR	Air Force Office of Scientific Research
AFRL	Air Force Research Laboratory
AFSCN	Air Force Satellite Control Network
AFSPACECOM	Air Force Space Command
AFSWC	Air Force Space Weather Center
AFWA	Air Force Weather Agency
AFWIN	Air Force Weather Information Network
AF/XOW	Air Force Director of Weather
APL	Applied Physics Laboratory of Johns Hopkins University
ASCII	American Standard Code for Information Interchange
ASPAM	Atmospheric Slant Path Analysis Model
AVHRR	Advanced Very High Resolution Radiometer
AVN	Aviation Model
AVO	Alaska Volcano Observatory
BATS	Biosphere-Atmosphere Transfer Scheme
CLASS	Canadian Land Surface Scheme
CME	Coronal Mass Ejections
COE	Common Operating Environment
DII	Defense Information Infrastructure
DMSP	Defense Meteorological Satellite Program
Dst	Disturbance, storm
ECMWF	European Center for Medium-Range Weather Forecasts
EIT	Extreme Ultraviolet Imaging Telescope
EVA	Extravehicular Activities
FAC	Field Aligned Currents
FNMOC	Fleet Numerical Meteorology and Oceanography Center
FSL	Forecast Systems Laboratory
FTP	File Transfer Protocol
GDS	Grid Description Section
GI	Geophysical Institute
GIC	Ground Induced Currents
GIF	Graphic Interchange Format
GIT	Georgia Institute of Technology
GMT	Generic Mapping Tools
GOLD	Geophysical On-Line Data
GOES	Geostationary Operational Environment Satellite

GRIB	Gridded Binary
GSE	Geocentric Solar-Ecliptic
GSFC	Goddard Space Flight Center
GUI	Graphical User Interface
HLBL	High Latitude Boundary Layer
IDL	Interactive Data Language
IMF	Interplanetary Magnetic Field
ISS	International Space Station
JHU	Johns Hopkins University
JHU/APL	Johns Hopkins University/Applied Physics Laboratory
Kp	Planetary Index of Geomagnetic Activity
LAN	Local Area Network
LAPS	Local Analysis and Prediction System
LASCO	Large Angle Spectroscope Coronagraph
LEO	Low-attitude Earth Orbit
LSM	Land Surface Model
MATCH	Model of Atmospheric Transport and Chemistry
MeV	Million Electron Volts
MM5	Fifth Generation Mesoscale Model
NASA	National Aeronautics and Space Administration
NCAR	National Center for Atmospheric Research
NCEP	National Centers for Environmental Prediction
netCDF	Network Common Data Form
NGDC	National Geophysical Data Center
NGM	Nested Grid Forecast Model
NOAA	National Oceanic and Atmospheric Administration
NOGAPS	Navy Operational Global Atmospheric Prediction System
NRL	Naval Research Laboratory
NWP	Numerical Weather Prediction
NWS	National Weather Service
OWS	Operational Weather Squadron
PACE	Polar Anglo-American Conjugate Experiment
PBL	Planetary Boundary Layer
PCA	Polar Cap Absorption
PDS	Product Definition Section
PFRR	Poker Flat Research Range
PNG	Portable Network Graphics
PUFF	From Puff the Magic (ash spewing) Dragon

RBE	Radiation Belt Environment
SAA	South Atlantic Anomaly
SABER	Sounding of the Atmosphere using Broadband Emission Radiometry
SD	Space Department of the Applied Physics Laboratory
SDP	Software Development Plan
SEC	Space Environment Center
SEE	Solar EUV Experiment
SEON	Solar Electro-optical Observing Network
SEP	Solar Energetic Particles
SFOC	Space flight Operations Center
SOHO	Solar and Heliospheric Observatory
SPE	Solar Particle Event
STP	Solar Terrestrial Physics
SWOC	Space Weather Operations Center (Offutt)
SWXS	Space Weather Squadron
SXI	Soft X-ray Imager
Tcl	Tool Command Language
Tk	Toolkit
Tix	Tk Interface Extension
UAF	University of Alaska, Fairbanks
UCAR	University Corporation for Atmospheric Research
UCB	University of Colorado, Boulder
UPOS	University Partnering for Operational Support
UTC	Coordinated Universal Time
WDC	World Data Center
WF	Weather Flight
WMO	World Meteorological Organization
XDR	External Data Representation