Naval Research Laboratory

Washington, DC 20375-5320

NRL/MR/6003--19-9872

Airborne Puff and Plume Datasets for an Urban Landscape

Jay Boris
M.Y. Obenschain
Keith Obenschain

Laboratories for Computational Physics and Fluid Dynamics Branch
Material Science & Component Technology Directorate

April 18, 2019

DISTRIBUTION STATEMENT A: Approved for public release; distribution is unlimited.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing this collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. **PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.**

1. REPORT DATE *(DD-MM-YYYY)* 18-04-2019	**2. REPORT TYPE** Memorandum Report	**3. DATES COVERED** *(From - To)*

4. TITLE AND SUBTITLE Airborne Puff and Plume Datasets for an Urban Landscape	**5a. CONTRACT NUMBER**
	5b. GRANT NUMBER 64-1G56-0-9-5
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Jay Boris, M.Y. Obenschain, and Keith Obenschain	**5d. PROJECT NUMBER**
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) U.S. Naval Research Laboratory 4555 Overlook Avenue, SW Washington, DC 20375-5320	**8. PERFORMING ORGANIZATION REPORT NUMBER** NRL/MR/6003--19-9872
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)	**10. SPONSOR / MONITOR'S ACRONYM(S)**
	11. SPONSOR / MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

DISTRIBUTION STATEMENT A: Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

This report describes a database of ”Puffs and Plumes” containing four high-resolution CFO datasets and associated files that capture he turbulent, time-dependent downwind evolution of ground-level tracer-gas clouds in a fictitious urban environment. Time histories of clouds from both continuous sources (plumes) and instantaneous sources (puffs) are presented. The database also contains files defining the urban geometry, a mask to overlay a 2D visualization of the geometry for display, and Fortran utilities to read and test the our compressed datasets. These datasets contain high-frequency time sequences of ground-level, neutrally buoyant, gas tracer density ‘ields computed at 5-meter resolution on a uniformly spaced 1200x800 computational grid. These density fields, for both the “puff” and “plume” sources, originate from six separate locations in the 6-km by 4-km urban domain. These datasets are structured to allow rapid analysis of turbulent fluctuations in the concentration of evolving puff and plume clouds and to study the naturally occurring variability expected between distinct realizations of instantaneous releases in fluctuating wind fields.

15. SUBJECT TERMS

Airborne contaminant datasets	Puffs and Plumes
Urban landscape	High resolution data

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** Jay Boris
a. REPORT Unclassified Unlimited	**b. ABSTRACT** Unclassified Unlimited	**c. THIS PAGE** Unclassified Unlimited	Unclassified Unlimited	52	**19b. TELEPHONE NUMBER** *(include area code)* 202-767-3055

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39.18

This page intentionally left blank.

Abstract

This report describes a database of "Puffs and Plumes" containing four high-resolution CFD datasets and associated files that capture the turbulent, time-dependent downwind evolution of ground-level tracer-gas clouds in a fictitious urban environment. Time histories of clouds from both continuous sources (plumes) and instantaneous sources (puffs) are presented. The database also contains files defining the urban geometry, a mask to overlay a 2D visualization of the geometry for display, and Fortran utilities to read and test the four compressed datasets. These datasets contain high-frequency time sequences of ground-level, neutrally buoyant, gas tracer density fields computed at 5-meter resolution on a uniformly spaced 1200x800 computational grid. These density fields, for both the "puff" and "plume" sources, originate from six separate locations in the 6-km by 4-km urban domain. These datasets are structured to allow rapid analysis of turbulent fluctuations in the concentration of evolving puff and plume clouds and to study the naturally occurring variability expected between distinct realizations of instantaneous releases in fluctuating wind fields. They can be used to assess the errors made in using time-steady models and other simplified representations of the urban fluid dynamics and they can be used to simulate proposed sensors and sensor systems in realistic airborne contaminant flows.

1. Introduction and Background

These four datasets were assembled from extensive CFD runs over the typical but fictitious urban domain shown in Fig. 1 below. FAST3D-CT, NRL's detailed Contaminant Transport (CT) model, was used to compute these databases. Field trials generally provide only a single realization of the flow and corresponding tracer cloud structure. However, in urban situations the difference between distinct realizations of the flow, determined for example by beginning each computer simulation (or field measurement) at a different time, can be quite large. Even wind tunnel campaigns, which can provide many independent realizations, have difficulty going back and adding more data or refining a measurement after the limited-duration experimental campaign is complete. High-resolution, time-dependent, Large Eddy Simulation allows a class of scenarios to be revisited as often as desired to answer specific questions or to improve the statistics by collecting more data.

Detecting and understanding airborne contaminants, reducing risks from Chemical, Biological, or Radiological (CBR) agents released into the air, and a fundamental interest in turbulence and complex fluid dynamics motivate our interest in the evolution and fluctuations of plumes and puff sources of airborne contaminant. Computing contaminant transport (CT) accurately across a typical urban geometry is a difficult, complicated problem (e.g. Boris, 2002, 2005; Britter and Hanna, 2003; Patnaik, et al., 2009; Fischer et al., 2012; Leitl, et al., 2013, 2015, and 2016).

Ultimately, however, the fluid dynamics of airflow through a city controls the transport and dispersion of airborne contaminants. It also controls the environmental air quality, the wind forces on buildings, and the ambient noise level due to the winds. These are urban aerodynamics problems primarily, not meteorology. The space scales are short, a few meters to at most a few kilometers. The average airflow, the dynamic fluctuations, and the building-scale turbulence are all closely coupled to the complicated geometry. Urban aerodynamics is driven by a deep,

---

Manuscript approved February 26, 2019.

stratified urban boundary layer with significant wind fluctuations. Buildings create large "rooster-tail" wakes; they shed vortices dynamically and support complex recirculation zones. There are systematic fountain flows up the backs of tall buildings and dust in the wind can move perpendicular to or even against the locally prevailing wind direction. Solar heating casts shadows from buildings and trees while aerodynamic drag and heat exchange are affected by the surface property variations and turbulent heat transport.

We require time-dependent, three-dimensional Computational Fluid Dynamics (CFD) to predict these unsteady, obstructed, buoyant flows accurately and the dynamic contaminant plumes that they drive. In principle, meteorology drives the airflow in a city and can provide aerodynamic boundary conditions for an urban domain but the weather can be treated as known over times of a few minutes to a fraction of an hour. Urban airflow over large areas has shorter widely varying temporal and spatial scales that exhaust current modeling capacities. Crucial aerodynamic issues include turbulent fluid transport in complex geometry, interacting wakes, and modeling boundary conditions. These issues force us to pay close attention to the recurring themes of spatial resolution (cell sizes), flow unsteadiness, and thermal stratification.

Computing urban aerodynamics accurately is a time-dependent, High-Performance Computing (HPC) problem. Section 2 of this report describes FAST3D-CT, NRL's high-resolution Monotone Integrated Large Eddy Simulation (MILES) model, for computing this CT database. In typical urban scenarios most particulate and gaseous contaminants behave similarly with respect to the overall transport and dispersion but the full physics of multi-group particle and droplet distributions are required for some problems. Section 3, "Urban Geometry and Grid for the MILES Database Simulations," considers the urban geometry definition and finite-volume CFD grid used by FAST3D-CT to represent this geometry. This section also contains some details of the runs actually performed.

Section 4, "Continuous Source Datasets for Six Evolving Plumes," describes the dataset named *PLUMES_xsecs* (78.67 GB) that contains 11401 cross-sections ("xsecs") for six distinct plumes, each originating from a different continuous source location. Cross-sections are available every 2.5 seconds (25 timesteps) for almost 8 hours. It also describes the dataset named *PLUMES_tseqs* (78.17 GB) that contains exactly the same information as *PLUMES_xsecs* but in a transposed format that enables much faster access to entire time sequences ("tseqs") at chosen grid locations. Section 4 contains a subsection discussing and illustrating the test output of program READPLUMES. Short printouts allow a new user to verify that the dataset read utilities and datasets, when implemented on another computer system, return the same numbers.

Section 5, "Acute Source Datasets for Sixteen Realizations of Six Sources," describes the dataset named *PUFFS_xsecs* (51.39 GB) that contains 601 cross-sections ("xsecs") for each of sixteen realizations of puff clouds evolving from each of six distinct acute (instantaneous) sources at different locations. Cross-sections are recorded every 10 seconds for 100 minutes after puff release. Section 5 also describes the dataset named *PUFFS_tseqs* (43.56 GB) that contains exactly the same information as *PUFFS_xsecs* but in a transposed format that enables much faster access to entire time sequences ("tseqs") at chosen grid locations. Section 5 contains a subsection discussing and illustrating the test output of program READPUFFS as for READPLUMES above.

Sections 6 and 7 present "Typical Plume Examples and Results" and "Typical Puff Examples and Results." These primarily graphical examples are not analyzed in much detail but rather are chosen to illustrate the differences between acute puff clouds, which dissipate in the vicinity of the source rather quickly, and plumes from continuous sources that build up a footprint of contamination that persists at least as long as the source continues. These examples also illustrate the importance of capturing the details of naturally occurring fluctuations and turbulence in time-dependent tracer concentrations beyond the capability of steady state and RANS models to provide. These sections highlight a range of interesting scientific issues with unknown or even incorrectly know answers where this database might be employed.

Section 8 contains a short summary and is followed by three appendices. Appendix A contains the source code for five Fortran subroutines that we are providing so users can easily access the four datasets, which have been compressed using the 'lz4' algorithm. These subroutines are wrappers to the 'lz4' routines that actually read and decompress the data in the four datasets comprising the "Puffs and Plumes" database. Appendix B contains the Fortran code for our READPLUMES test/verification program and Appendix C contains the corresponding READPUFFS program for the two puff datasets. These programs not only provide examples of how to call the interface utilities of Appendix A, they introduce some possible applications.

2. FAST3D-CT: NRL's Detailed Contaminant Transport Model

Direct numerical simulation (DNS) is the time-dependent solution of the full Navier-Stokes equations down to small scales where turbulence is dissipated by molecular viscosity. DNS is out of the question for simulations of airflow over cities using today's computer technology. Standard industrial approaches, such as Reynolds Averaged Navier-Stokes (RANS) models, occupy the other end of the CFD fidelity spectrum. These models simulate the mean flow and approximately model the turbulence but cannot capture important aspects of the inherently turbulent plume dynamics being driven by the urban geometry. Large Eddy Simulation (LES) constitutes an effective intermediate approach between DNS and the RANS and other steady-state methods (Sagaut, 2004, Grinstein et al., 2010). LES captures flow features that cannot be seen in time-averaged (steady-state) methods, such as significant flow unsteadiness and large contaminant density fluctuations, and provides higher accuracy than the industrial methods at an appreciably lower cost than DNS. Furthermore, LES solutions converge to the solutions of the Navier-Stokes equations as resolution is increased while less costly methods generally do not. The LES approximation is the proper class of models to adopt to study airflow fluctuations.

NRL's FAST3D-CT Monotone Integrated LES (MILES) model was used in the computing the detailed simulation datasets presented here. Previous applications of FAST3D-CT have demonstrated that detailed CFD, in the LES approximation, can be executed for urban aerodynamics with limited computational resources at the building and street scale (Boris, 2002; Patnaik and Boris, 2005; Boris, *et al.,* 2009, 2010). FAST3D-CT has been in development, validation, and use for more than 20 years (e.g., Boris, et al., 1992; Young, et al., 1993; Boris, 2005; Harms, et al., 2007, 2011; Patnaik, et al., 2009; Hertwig, et al., 2016a, 2016b; Boris and Patnaik, 2014, 2019; Boris, et al., 2019) described FAST3D-CT in considerable detail and introduced its use as the foundation for an operational CT-Analyst model. These articles and the book chapters by the NRL team (e.g., Patnaik, et al. 2005, 2007) provide detailed information about FAST3D-CT for urban aerodynamics with inclusive references. Buoyancy is handled in a potential-temperature formulation using a solar-heating ray-trace algorithm where buildings and trees can cast shadows.

FAST3D-CT is based on a scalable (parallel), low dissipation, 4th order phase-accurate Flux-Corrected Transport (FCT) convection algorithm. It uses the nonlinear, monotonicity-preserving

Flux-Corrected Transport (FCT) convection algorithm (Boris, 1971; Boris and Book, 1973, 1976). Young, et al. (1993) describe FCT and the treatment of complex geometry underlying FAST3D-CT. Kuzmin, et al. (2005) survey and analyze progress in FCT algorithms worldwide.

The Monotone Integrated LES (MILES) turbulence model (Boris 1989; Boris, et al. 1992) is implicitly included in FAST3D-CT. MILES' efficiency is well matched to CFD-based plume simulation at the urban scale, an application where classical LES methods are more expensive. A typical run with the FAST3D-CT model, such as the 1200 x 800 x 100-cell computations performed for these puff and plume datasets, takes about 10 hours on a single 24-core OpenMP node, for 30,000 timesteps. This is significantly faster than classical CFD models due to the savings achieved by MILES as well as other algorithmic improvements. See the book ***Implicit Large Eddy Simulation: Computing Turbulent Flow Dynamics*** (Grinstein et al., 2010) for detailed discussions and comprehensive validation studies of this MILES approach.

In FAST3D-CT the wind boundary conditions are forced to fluctuate. A deterministic model for a synthetic, non-periodic realization of these input fluctuations is used with three free parameters, one for scaling fluctuation size, one for strength, and one for time scale. These inputs allow users to approximate different types of atmospheric conditions. This fluctuating wind model, an analytic function defined throughout the computational domain, has several physics-based components. It serves as initial conditions at the beginning of a run and is updated thereafter at the sides and top of the domain as evolving boundary conditions. All fluctuations occurring within the domain during a simulation are naturally sustained by the vortices shed from buildings and the nonlinear evolution of the impressed fluctuations at the boundaries as they are convected through the domain. When the composite, impressed turbulent field at the boundaries flows over 500 to 1000 meters of actual city geometry, initial errors and inconsistencies give way to a geometry-dependent, self-consistent flow. See Boris (2005) for a more extensive discussion. The timestep was held fixed at 0.1 sec per step for all simulations.

The FAST3D-CT model also has physics-based algorithms and input for important droplet and aerosol physics including evaporation, deposition, and re-lofting. The model can be run in a single-phase (vapor) mode, in a two-phase (vapor and particles or vapor and droplets) mode, or in a three-phase (vapor, particles, and droplets) mode. FAST3D-CT is a "multi-group, multiphase" model in these applications. Since FAST3D-CT originated as a reactive flow code, each scenario can have up to 12 separate species and they can react chemically. Most of these additional terms are not used in the runs collected here and thus the significant problem of what physical numbers should be used for all these effects is not an issue.

Uncertainty, Variability, and Fluctuations

"Uncertainty" describes the situation when quantities are simply unknown. "Variability" applies when the particular values assigned to a quantity at different times and places appear random but these values are being drawn from a known statistical distribution. What chemical was spilled and when it was spilled may be *uncertain*. On the other hand, the atmospheric turbulence, the gusts in the wind, and all related fluctuations can be measured in almost any environment, at least in principle, so they are considered variable but not really uncertain. In practice the wind variability in an urban setting is seldom measured accurately enough to be the basis for validating CFD models although appreciable information is sometimes available.

Even if the wind variability is very well characterized, however, it does not follow that the corresponding variability in contaminant concentrations measured at some location far from a source can be known. The downwind contaminant concentration is determined by a complex, nonlinear continuity equation that integrates over the complete time- and space-dependence of the air velocity. The concentration value at a point, therefore, has many possible values depending on the particular realization of the airflow time history that is chosen from the ensemble of possible flow solutions corresponding to the current environmental conditions. Often field trials have emphasized long duration releases and long duration sampling intervals to reduce measurement variability and thus mask the roles of gusts and fluctuations.

Results from the datasets in this "Puff & Plume" database show that the differences from one realization to another are substantial even for releases of extended duration. The wind fluctuation and vortex-shedding phase differences between realizations account for the variety of observed plume and cloud shapes and concentration distributions (e.g. Figs. 4 and 6 discussed below). Contaminated areas can differ by factors of two or more. We must conclude that there is no single "true" (experimental or theoretical) answer to be compared with simulations or models. Measuring multiple realizations for fixed environmental conditions is usually not possible at city scale in a field trial, leaving us with no quantitative yardstick to compare experimental and computed concentration values (Obercampf and Helton 2002; Oberkampf and Barone 2004). Wind tunnels and CFD simulations must be used to fill this gap.

3. Urban Geometry and Grid for the MILES Database Simulations

Figure 1 below shows a 6 km by 4 km urban region where the building outlines, which are taken from a 1-meter resolution digital terrain map, are contoured at ground level. Yellow circles, labeled S1 to S6 in the figure, show the locations of six separate ground-level sources, some near buildings and some in relatively open areas. NRL's FAST3D-CT MILES simulation model, described above, was run in this geometry to advance six independent tracer species simultaneously using a 5-meter spatial resolution grid with a 3 m/s wind from 300^0 (from 30^0 north of west).

The six source locations were chosen to sample different local geometry conditions. Source locations S2 and S3 are directly adjacent to or embedded in the buildings. Source locations S1 and S5 appear more open than S2 and S3 but even in these cases, particularly S5 illustrated in Fig. 17 below, the nearby geometry plays a qualitatively important role when combined with the strong fluctuations in wind strength and direction. Source locations S4 and S6 were chosen to highlight the effects of urban geometry in dispersing the tracer laterally and trapping tracer for long times where it otherwise might blow away rather quickly. The single wind direction of 300^0 was chosen for the Plume and Puff datasets because this direction, given the placement of the sources, allows the tracer plumes and clouds to spend a large amount of time before exiting the computational domain.

The average wind velocity and temperature profiles were provided by NRL's COAMPS-OS model (Holt, *et al.,* 2009, 2011) for conditions near neutral buoyancy that suppress rapid vertical dispersion of the tracer. Figure 18, discussed in the summary, shows the particular temperature and velocity profiles used as inflow and boundary conditions for these runs. Chaotic, non-periodic fluctuations were imposed on the average profiles at the inflow and top boundaries using a model developed for FAST3D-CT and validated through the OKC field trials and in the University of Hamburg wind tunnel. This model is described briefly in Section 2 above.

The continuous-source plume databases described in Section 4 were extracted from a run of more than 8 hours real time (300,000 timesteps of 0.1 sec duration) while the acute, 16-realization, "instantaneous puff" databases described in Section 5 were extracted from a run of more than 27 hours real time (975,000 time steps). In this report we will identify the tracer clouds from continuous sources as "plumes" and the tracer clouds from acute sources as "puffs."

Both plumes and puffs fluctuate vigorously in the self-consistent time-dependent winds due to natural wind gusts and building vortex shedding. The first 15,000 steps in these runs were used to “spin up” the fluid dynamics. The initial releases all begin at step 15,001.

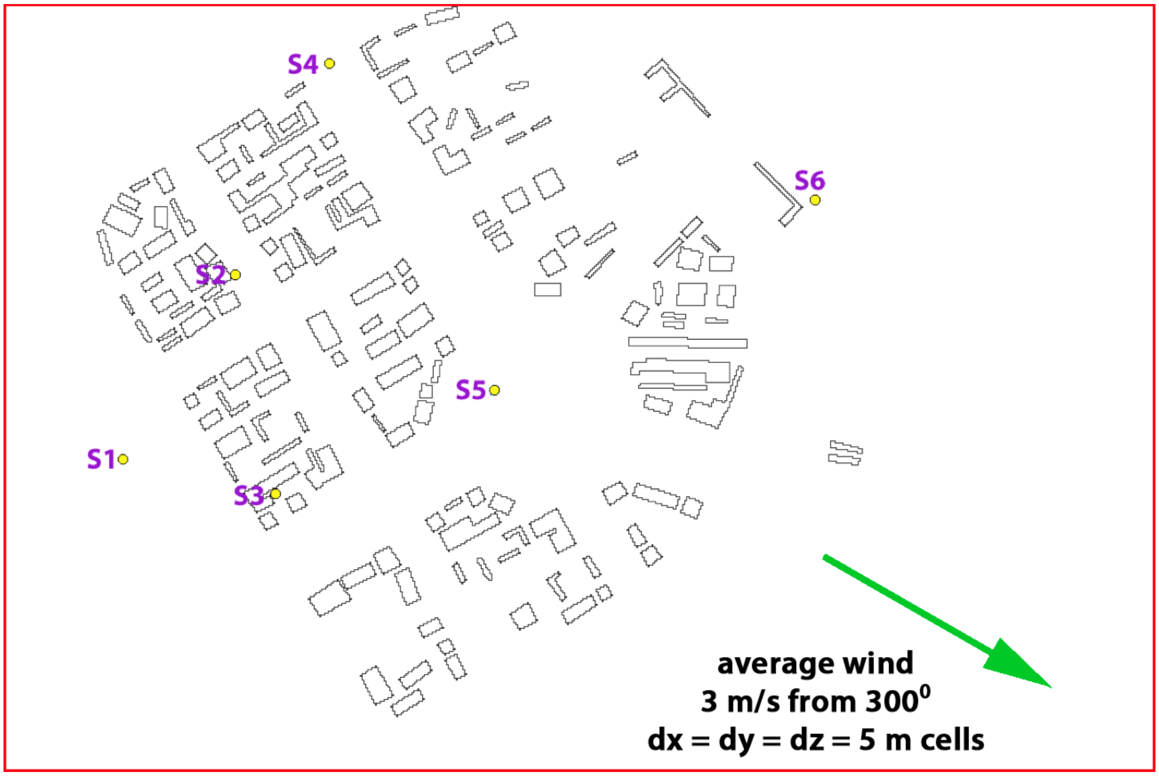

Figure 1. Continuous (steady) sources and acute (instantaneous) sources lay down a gaseous tracer at the 6 locations indicated above in a 6 km by 4 km urban domain. The wind is from 300^0 at 3.0 m/s and the temperature is 0^{0}C. The cell size for the FAST3D-CT simulation was 5 meters.

Figure 2 shows the buildings for this domain colored by their height. The darkest red building has its flat top at 281.0 m above the reference level. These are not building heights above ground, a measure that must be computed by subtracting the ground height (Fig. 3 below) from the building height at the same location. This difference depends on which side of the building the height above ground is being measured.

The light blue buildings in the lower center of the Fig. 2 have their tops about 50 meters above the reference level but only one or two stories above the ground (Fig. 3). The overall database contains an unformatted binary file named *urban_geometry* (7.7 MB). This file contains two real arrays, ‘Bhgt’ and ‘Ghgt,’ giving the building height and ground height at the center of each 5-meter cubed CFD cell on the NXxNY x NZ grid (1200 x 800 x 100). Cells near the top are stretched extending the vertical grid to 1.5 km. A shallow but stabilizing temperature gradient was chosen to force the tracer to stay near the ground without rapid vertical dissipation.

This file is read in by the test programs so a prospective user can see how to acquire and work with the relevant geometry when necessary. The database also provides an NX by NY file of 2-byte signed integers named *urban_geometry_MASK* that marks the geometry pixel by pixel, as used in the first three figures, to indicate cells that are inside but at the edge of buildings. The mask uses integer value 6 to indicate transparency and 30 to mark the edges of building. For real cities we use additional integers to mark water, trees and shrubs, roads, aspects of land use, and even provide some salient map labels. However, in this fictitious urban region and corresponding geometry definition, there are no trees or vegetation to complicate the issue and the land use between buildings is assumed totally unvarying, smooth and impermeable.

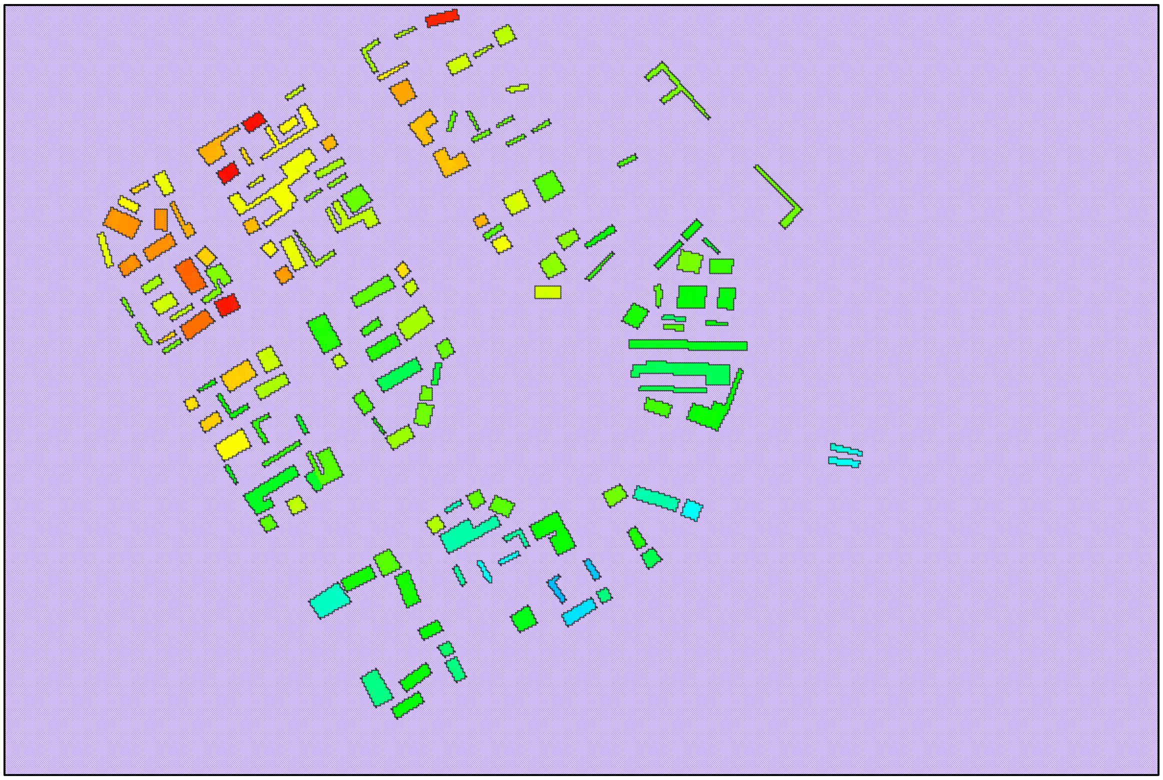

Figure 2. Building heights coded by color above a reference level set here at 0.0 meters. Red is 281 meters and dark blue is at the 0 meters reference level. In this urban geometry database the buildings are irregularly spaced and the terrain has a significant ground slope.

Figure 3 below shows the height of the ground above the reference level z = 0.0 m. The ground is color coded from 0.0+ m (dark blue) up to 220.5 m in the upper left (red). The pale lavender region in the lower right of Fig. 3 is flat and shows the reference level at 0.0 m. This occurs, for example, at a lakeshore or other waterfront. Near the right edge of Fig. 3, just above the lavender reference level, there are two diagonal rills in the ground whose berm edges are indicated by slightly lighter blue color. These ground features have a noticeable effect in some density plots (particularly for Source 6). Note that the ground definition is continuous right through the buildings. The buildings array is set at the reference level between the buildings, not to the ground height there. This allows the building array to indicate unequivocally where the buildings actually are. In the test programs, densities are set to -1.0 (an example) so that plotting programs can know why a density value is hard zero.

FAST3D-CT writes out the density of each tracer species from each source at specified intervals and at specified levels above the ground plane in the urban geometry definition. In this 1200 x 800 x 100 domain, a file containing one horizontal cross-section at each selected time for each of the 6 sources takes up 23.04 MB. The ground-level tracer-density values are the only ones recorded in the interest of keeping the datasets easily manageable in size and storage. These cross-sections were originally compressed without loss in FAST3D-CT by ‘gzip’ and took up to 14.7 MB for a single, hard-to-compress file. The datasets described in therse two sections were subsequently converted to ‘lz4’ compression because ‘lz4’ gains about another factor of two over ‘gzip’ for the type of data being archived here. ‘lz4’ is described and even available from a number of sources on the internet such as http://code.google.com/p/lz4/. These routines are included in the packaged ”Puffs and Plumes” database but not printed in this report.

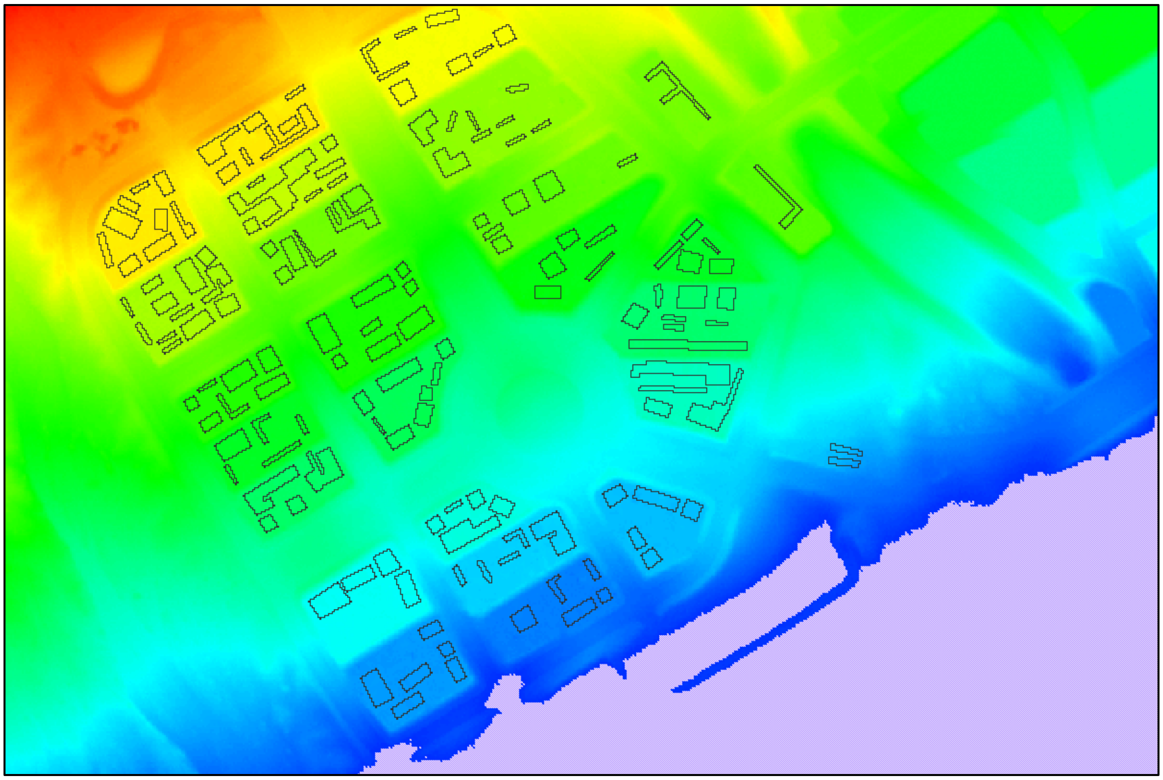

Figure 3. The ground elevation above the reference level 0.0 meters is color-coded with red indicating 220.5 meters in the upper left. Lavender in the lower and center right identifies the reference level 0.0 meters. The wind is from 300^0 at 3.0 m/s and the temperature is 0^{0}C. The 3D cubical cell size for these FAST3D-CT simulation was 5 meters.

Sections 4 and 5 following describe the databases of 8-hour plume simulations, containing 11401 ground-level cross-sections (“xsecs”) for each of six distinct plumes originating from different continuous source locations, and puff simulations of sixteen separate 100-minute, instantaneous “puff” source realizations from each of the same source locations as the plumes.

4. Continuous Source Datasets for Six Evolving Plumes

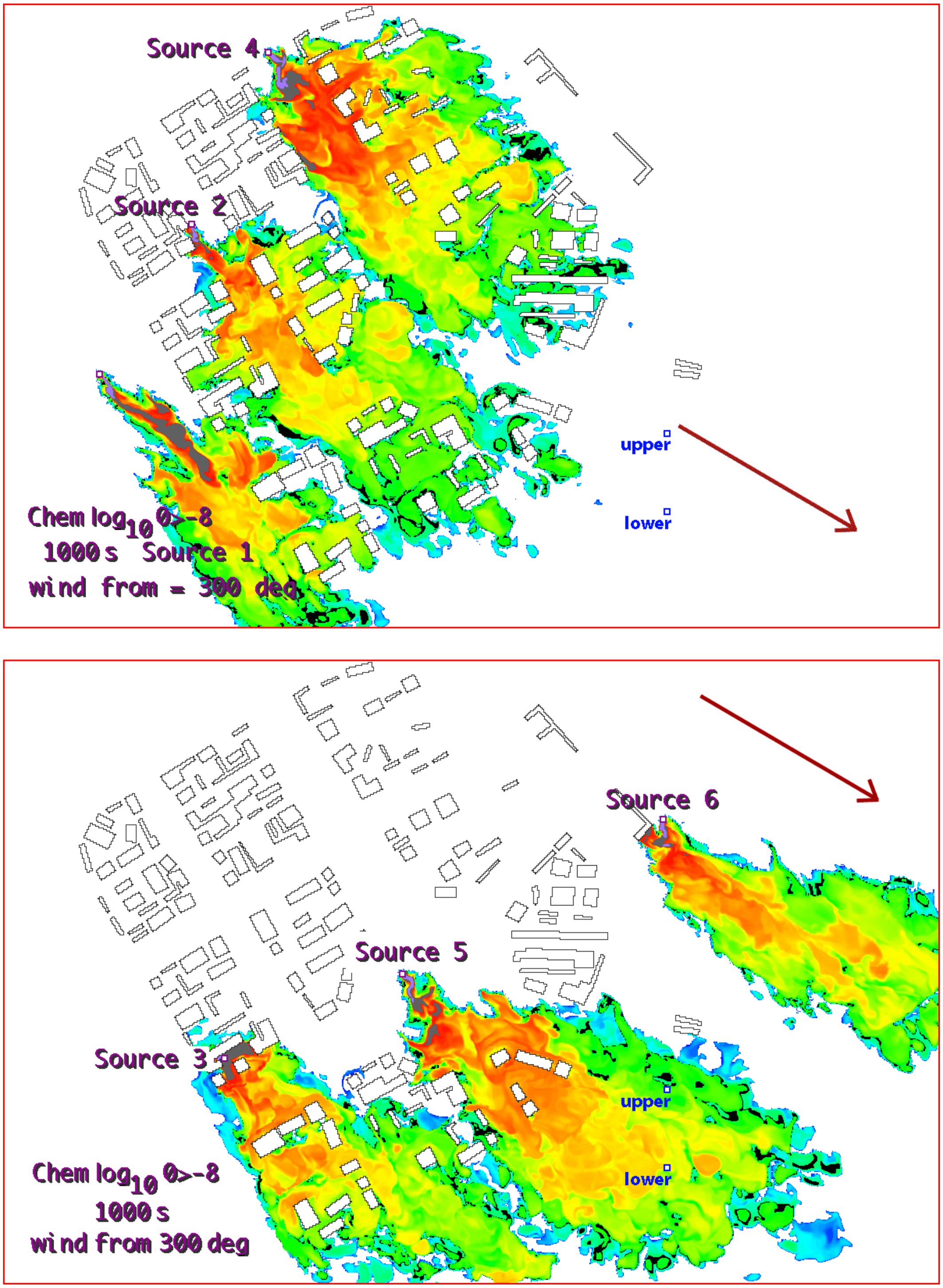

Figure 4. Density color contour snapshots of plumes evolving from 6 continuous sources. The color scale is logarithmic with red indicating 1 gm/m^3 and blue (white bordering blue) at 10^{-8} gm/m^3. Black indicates densities near 10^{-4} gm/m^3. Separate images have been overlaid at 1000s after the beginning of the 6 tracer releases. Variations of the plume shape due to trapping behind buildings and urban geometry are illustrated. Red arrows indicate the 3.0 m/s wind, from 3000.

Even when a tracer source is continuous and steady, the fluctuations in the driving wind make the resulting plume very unsteady. Figure 4 superimposes simple color-contour snapshots of the 6 plumes in the two panels above. These data planes from the *PLUMES_xsecs* dataset were recorded at time step 25,000, 1000 seconds after release begins at step 15,000, i.e. $\delta t = 0.1$ second. Separate density color contour plots from each source are superimposed in the upper and lower panels of the figure. At this early time the plumes do not overlap spatially. The figure illustrates the differences and variability that can be seen even with steady tracer release rates. The plume from Source 1 (lower left of the upper panel of Fig. 4) is emitted in an open area and thus is narrow near the source, not having been strongly dispersed by the influence of buildings until further downwind. As a corollary, because this particular plume volume is relatively small, the density near the source is also higher than the other five sources. The logarithmic color scale, used throughout this report has red indicating 1 gm/m^3 and the edge of the blue at 10^{-8} gm/m^3.

Our main interests have been in the extent, the distribution, and the temporal spectrum of the density fluctuations, and in the intermittency of the flow as reflected in the time-varying tracer density. The tracer density can be scaled to a wide range of source strengths as long as the mass density is low enough that buoyancy is unimportant. To make these datasets, a CFD spin-up interval of 15,000 timesteps (25 minutes) was allowed to elapse before the six sources zre switched on. Each one thereafter emits 100 kg per second of passive (neutral) buoyancy tracer. This release rate was continued throughout the duration of the run, almost 8 hours of real time (almost a week of computing). The mass density from the six separate tracer plumes was recorded across the entire domain at ground level every 25 timesteps (2.5 seconds) for 300,000 timesteps. The resulting two datasets, *PLUMES_xsecs* and *PLUMES_tseqs*, allow a number of analyses to be performed.

4.1. Plume Cross-Section Dataset *PLUMES_xsecs:*

A single folder named *PLUMES_xsecs* (78.67 GB) contains all 11401 tracer density ground-level cross-section files for the continuous plume database. Each of these files contains NX x NY x Nsrc 32-bit real numbers in little endian format (23.04 MB), i.e. one ground-level plane for each source per file. The data are recorded in the center of the first cell above ground level ($z \sim 2.5$ meters) and were written out using *lz4* compression with the smallest files compressed to 94 KB (essentially all zero) and the largest compressed to 7.5 MB. Because these plumes do not dissipate, the average file size is 6.9 MB with an average compression ratio of 3.3 Fast read utilities are provided to decompress each file automatically as it is read. The files are named *plumes_6srcs_nnnnnn.lz*. Each file contains the six 1200 x 800 plume cross-sections at timestep *nnnnnn*. The utility subroutine READ_PLUME_XSECS, printed in Appendix A, calls a well-documented Fortran routine lz4.f, which, in turn, calls four packages that can be downloaded from the web, routines lz4.c, lz4hc.c, lz4.h, and lz4hc.h.

4.2. Transposed Time Sequence Dataset *PLUMES_tseqs:*

The ground-level density cross-sections in *PLUMES_xsecs* are also presented in a transposed format prepared by an auxiliary program. Thus analyzing entire time sequences in detail for one or two sources or a few grid points does not require reading and decompressing the entire *Plumes_xsecs* dataset. A folder named *PLUMEs_tseqs* contains six subfolders, *src01_seqs* through *src06_seqs*, one for each of the 6 sources/plumes. Each of these subfolders contains 2400 files, a single file for each 20x20 block of 5m cells in the 1200 x 800 urban domain. Each of these block files, named for example, *Source01_0020_640.lz* for the particular 20 x 20 block with CFD cell indices i = 1, …, 20 and j = 621, 640, is dimensioned 11401 x 20 x 20. The utility subroutine READ_PLUME_TSEQS, also printed in Appendix A, is provided to read these blocks of time sequences.

4.3. Discussion of READPLUMES Test Program Printed Output:

The ***Puffs and Plumes*** database contains two Fortran programs for validating installation and retrieval of data, one, named READPLUMES, for verifying the two plume datasets and one, named READPUFFS, for verifying the two puff datasets. The entire expected output files are included with the datasets so users will get a quick indication that they are indeed getting the data we and they expect. Here we briefly explain the different sections of output from program READPLUMES. Selected sections of the output are numbered below and will be described separately. The entire printed output file is included in the database.

```
   READPLUMES test program beginning execution:

1. READPLUMES input: NX    NY    Nsrc Ntms stepdel   delta_x delta_t dt_phys
                    1200   800     6 11401     25       5.000   0.100   2.500
```

The input dimensions of the various arrays comprising the database and the physical parameters controlling the grid and time stepping are echoed. The definitions and restrictions are the same as given below for the READPUFFS test program described below. Because we computed no separate plume realizations, rather one very long run, the interval 'stepdel' between time slices is 25 steps (2.5 seconds) to allow relatively high-frequency analyses of the fluctuations. The dataset includes steps 15,000 (after the CFD spinup) through 300,000 in intervals of 25 steps. This gives 11401 time slices.

```
2. READPLUMES mask file:urban_geometry_MASK  size NX x NY  1200  800
```

A geometry mask file, named *urban_geometry_MASK*, is included in the Puffs & Plumes database. It can be used with packages to plot the density, probability cross-sections and other analyses that would benefit from being superimposed on an image of the urban landscape. This mask file is a 1200 x 800 array of 16-bit integers that marks the pixels for the building outlines, as shown in Figs. 1, 2 and 3 above. There are only two values in this file. The transparent pixels are marked as 6 and the building edge pixels are marked as 30. Program READPUFFS shows how to read this file.

```
3. Urban_geometry test results ...
   Building max & min 281.00m   0.00m   Ground max & min 220.50m   0.00m
   #Bldg 0: 898940  #Grnd 0: 152077  #Both 0: 152077  #Both NZ:  61060
```

A geometry definition file, named *urban_geometry* (7,680,008 bytes), is included in the database. It makes more complex analyses of the plume transport and dispersion possible. There are no trees or water in this urban domain so *urban_geometry* is an unformatted binary file with only two 1200 x 800 arrays containing 32-bit real values (units meters) for the height of the buildings (and ground) at each i,j cell center above a reference level zero. As shown by Fig. 3 above, the southeast section of the domain is flat at the reference level. READPUFFS shows how to read this file and also performs a few simple counts to verify the geometry was read correctly.

The highest top of any building is 281 m above the reference level. This is one of the red buildings in the upper right of Figure 2. This building may not be the tallest because the height of a building is usually measured from the ground adjacent to the building. The ground, in this case, varies by 220.5 meters from the southeast to the northwest corner of the domain. As there are 960,000 cells in all and buildings cover only 61060 of these 5m x 5m cells (i.e. 960,000 - #Bldg 0), the domain is 93.6% open. The ground at the reference level in the southeast corner covers 152077 cells (#Grnd 0). Since this area has no buildings, both buildings and ground are zero over 152077 cells and both are non-zero in the 61060 cells occupied by buildings. Note that in this representation the ground is not zero inside buildings but the building array is zero between buildings.

```
4. Plume ground-level hazard areas (sq km)
     time      Plume 1    Plume 2    Plume 3    Plume 4    Plume 5    Plume 6
      0.00   0.000E+00  0.000E+00  0.000E+00  0.000E+00  0.000E+00  0.000E+00
    600.00   4.155E-01  4.126E-01  3.893E-01  6.115E-01  7.274E-01  5.447E-01
   1200.00   1.627E+00  3.974E+00  2.333E+00  3.966E+00  4.099E+00  1.791E+00
   1800.00   2.849E+00  7.412E+00  3.848E+00  1.062E+01  5.696E+00  2.132E+00
   2400.00   4.312E+00  9.121E+00  4.820E+00  1.317E+01  6.270E+00  2.148E+00
   3000.00   4.856E+00  9.977E+00  5.246E+00  1.387E+01  6.156E+00  1.888E+00
   3600.00   4.202E+00  9.616E+00  5.098E+00  1.430E+01  6.148E+00  2.114E+00
   4200.00   4.060E+00  9.625E+00  5.251E+00  1.493E+01  6.226E+00  2.206E+00
   4800.00   4.623E+00  9.730E+00  5.534E+00  1.532E+01  6.423E+00  1.972E+00
   5400.00   4.348E+00  9.903E+00  5.271E+00  1.542E+01  6.314E+00  1.942E+00
   6000.00   4.525E+00  1.001E+01  5.107E+00  1.529E+01  6.254E+00  1.908E+00
   6600.00   5.027E+00  1.011E+01  5.480E+00  1.549E+01  6.454E+00  1.960E+00
   7200.00   4.573E+00  9.984E+00  5.121E+00  1.549E+01  6.404E+00  1.918E+00
```

The evolving plume from each source covers some of the cells at each time slice but leaves many ‘uncontaminated’ cells at zero density. In the ‘Output 4’ example printed here, the number of contaminated cells is counted for each of the plumes and these counts are converted to ‘hazard areas’ with units of square kilometers. These hazard areas are computed and printed every 10 minutes for the first 2 hours of each release in the table above. One might expect that the plume hazard area would quickly reach an equilibrium size on a bounded domain and this is generally true but the area fluctuations can still be large late in time, as seen for Plumes 1 and 6.

```
5. Source cell indices from step  87000
   Source 1 is at cell  122  326  in FAST3D-CT
   Source 1 is found @  122  326  at step  87000  rhomax=  7.3056E+03
   Source 2 is at cell  240  518  in FAST3D-CT
   Source 2 is found @  240  518  at step  87000  rhomax=  1.3964E+03
   Source 3 is at cell  282  290  in FAST3D-CT
   Source 3 is found @  282  290  at step  87000  rhomax=  1.8552E+03
   Source 4 is at cell  338  738  in FAST3D-CT
   Source 4 is found @  338  738  at step  87000  rhomax=  3.3267E+03
   Source 5 is at cell  510  398  in FAST3D-CT
   Source 5 is found @  510  398  at step  87000  rhomax=  5.3157E+03
   Source 6 is at cell  844  596  in FAST3D-CT
   Source 6 is found @  844  596  at step  87000  rhomax=  3.3148E+03
```

In a continuous plume release the maximum density in the entire domain is in the vicinity of the source. When a source location is unknown this gives a way to find it. Verification test 5 just above compares the source locations used as input for the CFD run (first line for each source) and the location computed by this method (second line). The test program searches the entire domain, finds the location of the maximum density for each plume, and prints this information, along with the maximum density recorded there, in the short table above. Step 87000 is two hours into the run time.

```
6. Testing READ_PLUME_TSEQS for sources 1 to 5: output a .csv file
   upper cell (850, 250) found in block 43 13
   lower cell (850, 250) found in block 43  8
   Opened Upper&Lower_rhot.csv on unit 7
```

```
   time              Plume 1     Plume 2     Plume 3     Plume 4     Plume 5
   0.00  upper pt  0.000E+00   0.000E+00   0.000E+00   0.000E+00   0.000E+00
         lower pt  0.000E+00   0.000E+00   0.000E+00   0.000E+00   0.000E+00
 600.00  upper pt  0.000E+00   0.000E+00   0.000E+00   0.000E+00   0.000E+00
         lower pt  0.000E+00   0.000E+00   0.000E+00   0.000E+00   0.000E+00
1200.00  upper pt  0.000E+00   0.000E+00   0.000E+00   0.000E+00   6.248E-04
         lower pt  0.000E+00   0.000E+00   0.000E+00   0.000E+00   1.257E-01
1800.00  upper pt  0.000E+00   3.577E-05   0.000E+00   3.018E-03   6.017E-04
         lower pt  0.000E+00   5.636E-02   1.668E-07   1.463E-04   2.281E-01
2400.00  upper pt  0.000E+00   6.767E-03   0.000E+00   7.233E-02   4.554E-02
         lower pt  1.208E-05   1.443E-01   3.500E-04   5.152E-03   3.503E-01
3000.00  upper pt  0.000E+00   4.983E-03   0.000E+00   1.012E-01   7.633E-02
         lower pt  2.709E-07   9.808E-02   4.817E-05   6.106E-03   1.858E-01
3600.00  upper pt  0.000E+00   4.799E-04   6.465E-09   1.780E-01   2.059E-03
         lower pt  2.835E-10   2.445E-02   6.021E-06   4.355E-02   2.429E-01
4200.00  upper pt  0.000E+00   6.381E-03   6.798E-10   1.716E-01   4.253E-02
         lower pt  0.000E+00   5.224E-02   1.125E-05   1.915E-02   1.385E-01
4800.00  upper pt  0.000E+00   9.995E-04   0.000E+00   2.117E-01   3.932E-03
         lower pt  3.239E-08   8.488E-02   8.407E-05   2.544E-02   2.254E-01
5400.00  upper pt  0.000E+00   1.032E-04   0.000E+00   1.411E-01   9.020E-07
         lower pt  1.615E-06   7.378E-02   2.183E-05   1.270E-02   1.898E-01
6000.00  upper pt  0.000E+00   3.593E-03   1.267E-08   1.158E-01   9.092E-02
         lower pt  2.349E-06   7.597E-02   2.342E-05   3.081E-02   1.668E-01
6600.00  upper pt  0.000E+00   1.755E-02   3.070E-08   9.030E-02   3.208E-01
         lower pt  1.634E-06   1.822E-01   1.917E-04   6.234E-03   1.989E-01
7200.00  upper pt  0.000E+00   5.585E-03   2.137E-08   4.089E-02   5.295E-02
         lower pt  2.625E-08   6.018E-02   1.842E-05   1.030E-02   2.105E-01

READPLUMES: End of Computations
```

The final **READPLUMES** test, Number 6, involves reading and decompressing the plume time sequences in *PLUMES_tseqs.* Two measurement stations, chosen to be in the path of most of the plumes, are illustrated above in Fig.4 and are called the ‘upper’ and ‘lower’ points. Their positions differ by 500 meters in the north-south direction. These two stations are shown by blue squares around white in the two panels of Fig. 4 above. The i, j indices are (850, 150) for the “lower” and (850, 250) for the “upper” location in the CFD grid with the lower left corner of cell numbered (1,1) defining the origin of the X-Y Cartesian coordinate system. These points reside in two different 20x20 time-sequence blocks whose block indices are given above. Since the time-sequences for the separate plumes are stored separately, 12 calls to the utility READ_PLUME_TSEQS were needed to assemble the data for the table just above.

The density at the upper and lower points can be compared at 10-minute intervals (600 s) for the first two hours of the dataset in the short table above. Plume 1 never reaches the upper point although significant contamination of the lower point is seen from 40 minutes after release onward. Plumes 2, 3 and 4 all eventually contaminate both upper and lower points with first arrival around half an hour. Plume 5, whose source is closer to the two stations, arrives within 20 minutes while Plume 6 is far enough north that its tracer never arrives at either the upper or the lower points.

The READPLUMES test program also prints out a .csv (comma separated values) file for these 10 time sequences (upper and lower point densities for Plumes 1 through 5). These densities are recorded every 5 seconds for the first 2 hours of the release in file *Upper&Lower_rhot.csv*, a total of 1441 lines of data. Figure 5 immediately below plots the density-as-a-function of time traces from the 5 of 6 sources that had densities above the threshold 10^{-8}. The lines for the upper and lower measurement stations are colored similarly for each plume with the data for the lower point having a somewhat darker shade in each case. To further help distinguish these plotted curves the lower-point data have square markers added at the data points.

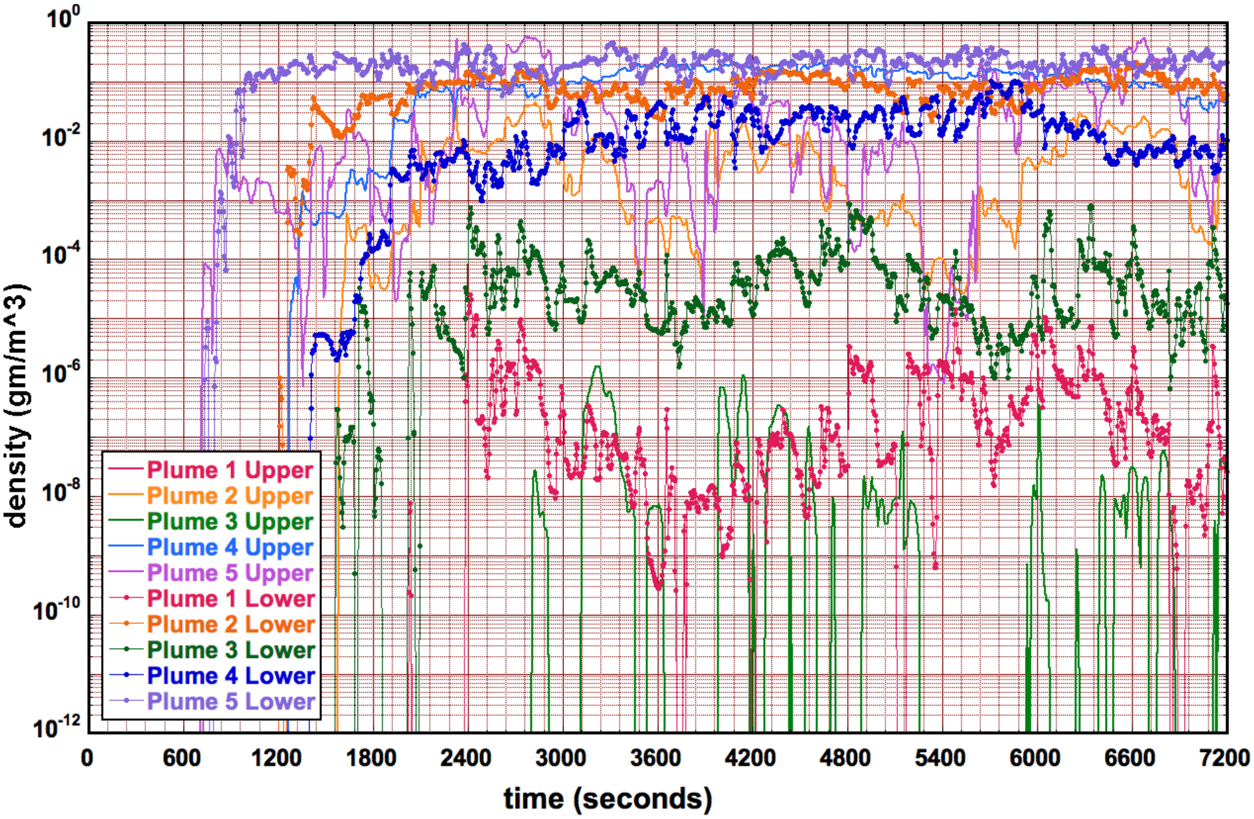

Figure 5. Tracer density in Plumes 1 through 5 at the upper and lower points as a function of time. Similar colors were chosen for the upper and lower traces of each plume with marker points used to distinguish the time sequences at the lower measurement station.

Consider first the two lavender curves for Plume 5. From Fig. 4 we can see clearly that the lower point (markers on curve) is near the center of Plume 5 and this is reflected by the high values of density in Fig. 5, varying only between 0.1 and 0.5 gm/m^3. The corresponding lavender curve without markers is for the upper point, which is closer to the northernmost edge of plume 5. The average density at this upper point is one to three orders of magnitude lower than in the center of the plume and the relative fluctuations are two orders of magnitude higher. Nevertheless, for Plume 5, the density does not seem to ever drop to zero.

Consider next the green curves for Plume 3. Source 3 is further from the measurement stations so the densities are considerably smaller than for Plume 5. In this case the lower point is also closer to the center of the plume but the fluctuations seen in the green curve with markers are considerably larger than for Plume 5. Furthermore, the density repeatedly drops to zero at the upper point for Plume 3, meaning that the edge of the plume intermittently and repeatedly sweeps back and forth across the measurement station location.

5. Acute Source Datasets for Sixteen Realizations of Six Sources

Many of the analyses illustrated above for “plumes” can be performed for tracer “puff” clouds as well with the added complication that even the ensemble-averaged solutions will be time dependent. There are two limiting cases for a source deposition time profile. The first, a constant release rate continuing forever, is captured in the *PLUMES_Xsecs* and *PLUMES_tseqs* datasets above. The second is an instantaneous “acute” or “puff” deposition where all of a tracer species is deposited at one time. Since the tracer subsequently can blow away with a decay time scale of an hour or so, the true long-time average is zero. Some other way must be found to normalize distributions. This also means that naturally occurring fluctuations cause identical releases from any location, which vary only in their release time, to differ greatly.

Acute sources, considered in this section, are important because accidents and deliberate acts are generally more serious when the contaminant densities are largest and because there is less time to respond. With acute “puff” sources the time dependence is even more severe than with plumes because the time scales are shorter and strong wind variability has no chance to even approximately smooth densities out. This variability is captured in datasets *PUFF_xsecs* and *PUFF_tseqs* by including 16 separate realizations of each instantaneous release and lasting for 1 hour 40 minutes in each case. These different realizations, one for each source location, were initialized every 60,000 timesteps and this are totally separate in time and uncorrelated.

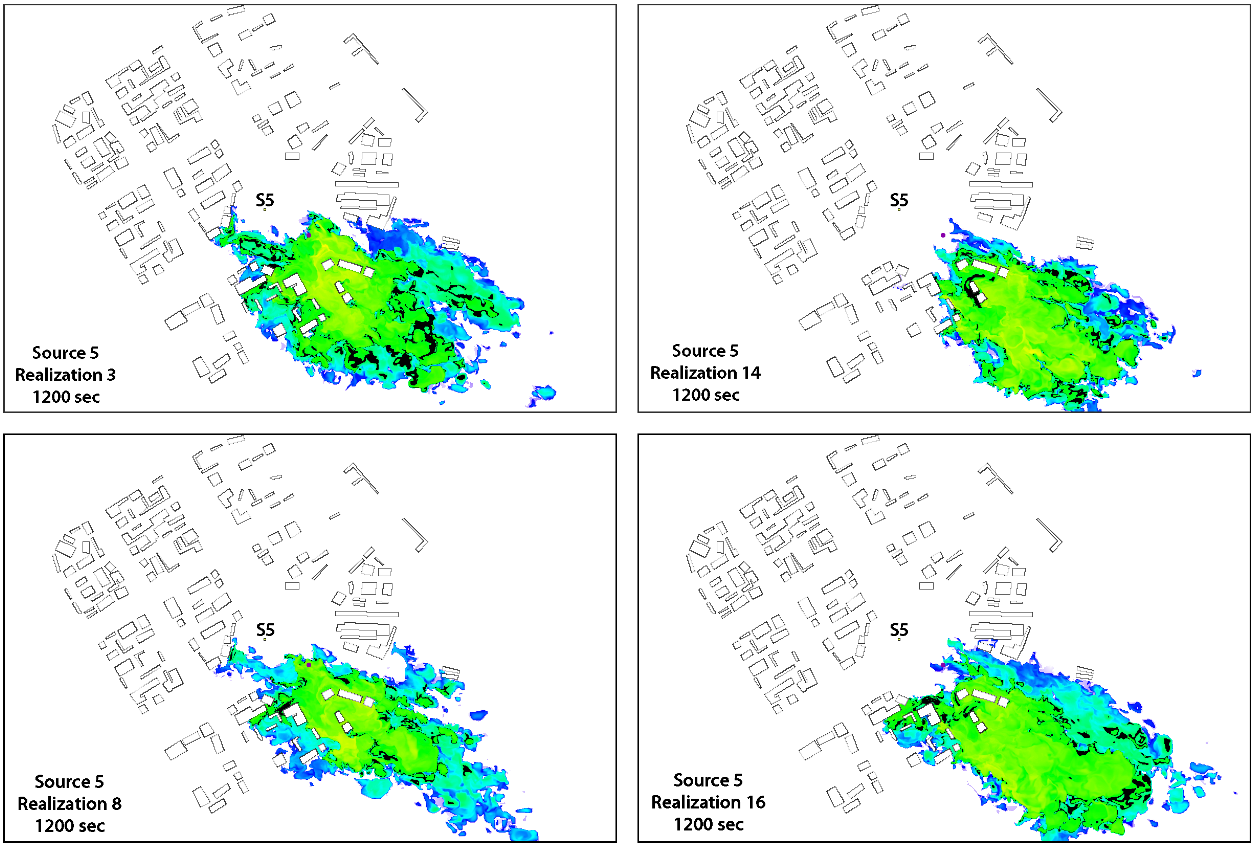

Figure 6. Four of the sixteen density cross-section realizations of the Source 5 puff release at 20 minutes. These “puff” clouds will drift rapidly away from the source in open locations so the variability and fluctuations are very large. Variations of the plume shape due to trapping of the tracer behind buildings and urban geometry are illustrated. The 3.0 m/s wind is from 300^0 (30^0 north of west).

Figure 6 shows a snapshot of the puff realizations numbered 3, 8, 14 and 16 from Source 5 captured 20 minutes after release. A purple circle, 500 meters southeast (down and right) of S5, marks the measurement station used for Fig. 7 below. Two of the sixteen realizations, shown on the left in Fig. 6, were chosen because they have a connecting trail of tracer trapped, partially upwind, behind the small building complex that is a couple of hundred meters south and west of the source. In the other two the cloud is moving away to the southeast with no sensible density connecting the cloud to the source region. This occurs when a particularly strong southward gust carries some tracer into the recirculation zone of these buildings early in the release before it can escape. This is important because it means that there is a lingering probability of contamination at points that are actually upwind of the source and quite in the open. Consequences of this are quite evident in Fig. 7 and in the time-dependent probability example of Fig. 17.

The datasets for acute (instantaneous) puffs was constructed from the ground-level (2.5-meter altitude) cross-sections from the six source locations but a finite amount of tracer, one metric ton, was released during a single time step at the beginning of each run segment. Two databases were constructed from the raw FAST3D-CT ground-level density cross-section files: 1) Database *PUFFS_xsecs* contains all *lz4* compressed ground-level density cross-sections (totaling 51.39 GB) and is contained in a single folder having 16 subfolders, one for each of the 16 realization of the 6 sources. In each of these subfolders there are 601 time slices (0 to 100 minutes after release) containing a density cross-section for all 6 sources for that time and realization. 2) Database *PUFFS_tseqs* contains 6 x 2400 files, each with a 20x20 block of density time sequences totaling 43.44 GB. *PUFFS_tseqs* has 6 subfolders, one for each source. Each of these subfolders contains files for the 2400 blocks of 20 x 20 cells spanning the computational domain.

5.1 Multi-Realization, Puff Cross-Section Dataset *PUFFS_xsecs*:

Consider the puff cross-section dataset. For each source (different source locations) the release occurred 16 times. The first realization subfolder in *PUFFS_xsecs* is labeled *PUFFS_rlz01* and totals 10.73 GB. It contains 2401 ground-level tracer density snapshots recorded every 25 timesteps from step 15000, the end of the CFD spin-up period, to step 75000. As the CFD time step was 0.1 second, there are 2.5 sec between snapshots for Realization 1 and this time history continues 6000 seconds after the instantaneous release. All 6 sources were computed simultaneously for each realization so the 6 density planes at each time are captured in a single file labeled, for example, *puffs_rlz01_015125.lz* for step 15125.

The interval between snapshots was extended to 100 timesteps (10 secs) for the next 15 realizations to reduce the total amount of data that had to be stored, and repeatedly re-read during each run. Thus each of these last 15 realization subfolders contains 601 compressed files and averages only 2.7 GB in total size. Each successive realization was computed after the previous realization had completed its 60000 timesteps, ensuring that the specific flow fields that were being used are totally independent with no overlap in time. This also means that the time-step numbers on the files extend up through 975000 as the last step of realization 16.

The individual files in *PUFFS_xsecs* are NX x NY x Nsrc in size, totaling 23.04 MB each. Since LZ4 compression works best when the data are unvarying (i.e. zero), these puff cross-section files range in size from 91KB (essentially empty) up to 6.6 MB in size. There are 11416 files with a total amount of data exceeding 263 GB. However, 'LZ4' compression, which is lossless, reduced storage by more than a factor of five down to 51.36 GB. Further, the LZ4 algorithm, decompressing the data while reading each file, takes much less time overall than just reading the full data. These compressed files are more than a factor of two smaller than the same information compressed by 'gzip.' In the plume cross-section database described just above, the individual LZ4 files ranged from 94 KB (essentially all zero) up through about 7.5 MB. The difference in maximum size between continuous and acute sources is caused by the increased amount of tracer in the system and the greater plume spread when the source is continuous.

5.2. Multi-Realization, Transposed Puff Time Sequence Dataset *PUFFS_tseqs:*

The ground-level density cross-sections in *PUFFS_xsecs* are also presented in a transposed format prepared by an auxiliary program. Thus analyzing entire time sequences in detail for one or two sources, a few realizations, or a few grid points does not require reading and decompressing the entire 51.39 GB dataset, *PUFFS_xsecs*. A folder named *PUFFS_tseqs* contains six subfolders, *src01_seqs*, …, *src06_seqs*, one for each of the 6 sources just as for the plumes time-sequence dataset. Each of these subfolders contains 2400 files, one for each 20 x 20 block of 5m cells in the 1200 x 800 urban domain. Each of these block files is named, for example, *Source01_0020_640.lz* for the particular 20 x 20 block with CFD cell indices i = 1, …, 20 and j = 621, …, 640. These individual blocks, however, differ from their plume dataset counterparts because each one contains all 16 realization of the puff. Each of these blocks is dimensioned 20 x 20 x 601 x 16. Users will have to be aware of this difference in programming applications. The utility subroutine READ_PLUME_TSEQS, printed in Appendix A, is provided to read these blocks of grid-point time sequences.

5.3 Discussion of READPUFFS Test Program Printed Output:

Following are selected portions of printed output from the **READPUFFS** verification program. These portions are numbered and will be described separately in the following. The entire printed output file is included in the database.

```
   READPUFFS test program beginning execution:

1. READPUFFS input: NX   NY   Nsrc Ntms Nrlz stepdel  delta_x delta_t dt_phys
                   1200  800    6  601   16   100      5.000   0.100  10.000
```

The input dimensions of the various arrays comprising the data based are given above. The definitions and restrictions are the same as for the READPLUMES verification program with the following change and addition. First, since there are no separate plume realizations tabulated, simply one long run, the interval *'stepdel'* between time slices steps was increased to 100 steps (10 seconds) to keep the overall size of the compressed 'puffs' datasets manageable with 16 separate realizations. Second the number of separate puff realizations has been added. For this ***Puffs and Plumes*** database, Nrlz = 16.

```
2. READPUFFS mask file:urban_geometry_MASK   NX x NY  1200  800
```

A geometry mask file, named *urban_geometry_MASK* (1,920,000 bytes) is included in the database. It may be used with packages to plot the density, probability cross-sections and other analyses that have to be superimposed on the urban landscape. This file is a 1200 x 800 array of 16-bit integers that marks the pixels for the building outlines, as shown in Figs. 1, 2 and 3 above. There are only two values in this file. The transparent pixels are marked as 6 and the building edge pixels are marked as 30. Normally we would use other integers to mark water, trees and shrubs, roads, aspects of land use, and some salient map labels. The programming in **READPUFFS** shows how to read this file.

```
3. Urban_geometry test results ...
   Building max & min 281.00m   0.00m   Ground max & min 220.50m   0.00m
   #Bldg 0: 898940  #Grnd 0: 152077  #Both 0: 152077  #Both NZ:  61060
```

The description of this urban geometry portion of the **READPUFFS** output is exactly as given above for plumes.

```
4. Bldg Zero, nonzero, and Open Zero, < threshold, and >= threshold counts:
   Source 1 #Bldg0 & NZ  61060    0   #Open0, <TH & >TH 898929    2     9
   Source 2 #Bldg0 & NZ  61060    0   #Open0, <TH & >TH 898897    7    36
   Source 3 #Bldg0 & NZ  61060    0   #Open0, <TH & >TH 898913    4    23
   Source 4 #Bldg0 & NZ  61060    0   #Open0, <TH & >TH 898892    8    40
   Source 5 #Bldg0 & NZ  61060    0   #Open0, <TH & >TH 898902    6    32
   Source 6 #Bldg0 & NZ  61060    0   #Open0, <TH & >TH 898923    4    13
```

The evolving cloud from each source covers some of the cells at each time slice but leaves many uncontaminated cells at zero density. In the Output 4 above, the number of building cells is the same for each source but the number of zero density and contaminated cells varies from source to source. This table is for realization 16 at time step 15100 (only 10 seconds after release) so the leading edge of the cloud has only moved a few meters. The numbers Open0 above are where the 6 source clouds have zero density. They cover almost the entire grid. The numbers Open>TH, generally two-digits, are the number of cells with a density greater than the value 'thresh' = 1.0E-8. The largest of these areas is only 1000 square meters (40 cells). The numbers Open<TH, indicate small areas having a density below 'thresh' but greater than zero. The density drop at the edge of the cloud is very sharp because Flux-Corrected Transport is a high-resolution algorithm and because only 100 time steps have elapsed.

```
5. Source cell indices from step  15100
   Source 1 is at cell  124  326  in FAST3D-CT
   Source 1 is found @  124  326  at step  15100  rhomax=  6.6753E+03
   Source 2 is at cell  242  518  in FAST3D-CT
   Source 2 is found @  242  518  at step  15100  rhomax=  3.0767E+03
   Source 3 is at cell  284  290  in FAST3D-CT
   Source 3 is found @  284  290  at step  15100  rhomax=  1.7944E+03
   Source 4 is at cell  340  738  in FAST3D-CT
   Source 4 is found @  340  738  at step  15100  rhomax=  1.4115E+03
   Source 5 is at cell  512  398  in FAST3D-CT
   Source 5 is found @  512  398  at step  15100  rhomax=  3.8631E+03
   Source 6 is at cell  846  596  in FAST3D-CT
   Source 6 is found @  846  596  at step  15100  rhomax=  3.2967E+03
```

Early in a puff release the maximum density in the entire domain is in the vicinity of the source. The test program searches the entire domain and finds the location of the maximum density for each source in realization 16, printing this information, along with the maximum density recorded there, in the short table above. The first of the two lines for each source gives the source locations used to initialize the release. These are from data statements in READPUFFS. The second of the two lines corresponding is computed using the maximum algorithm provided. Step 15100 is the first recorded time slice after the puffs are released at the end of step 15000 (for all realizations). A test with kr = 1 would allow performing this test at step 15025 but the source locations are the same because the puffs have not yet begun to decay.

The rest of the READPUFFS tests are based on reading and decompressing the time sequences in *PUFFS_tseqs*. Six sets of printouts are numbered 6, one for each plume. Each set shows results for a single cell 500 meters downwind of the corresponding source location. This cell, identified in the tables numbered 6, resides in a different 20x20 block of time sequences for each source. Two of these printed sets of verification tests are reproduced here, the one for Source 1 and the one for Source 5. Although each block file contains the time sequences for all 16 realizations, e.g. file *Source01_0220_280.lz* used for the output segment numbered 6 immediately below, the printed table contains verification densities for only the first 6 realizations.

```
6. Testing READ_PUFF_TSEQS for source 1
   icell,jcell indices  210  276   are found in 20x20 block ib,jb   11   14
   Bldg Zero, nonzero, and Open Zero, < threshold, and >= threshold counts:
   Source 1 #Bldg0 & NZ      0    0   #Open0, <TH & >TH  59453   958  9989

   Spot check density values for Source 1  in cell   210  276  Bhgt    0.00
   time(s)    rlz 1      rlz 2      rlz 3      rlz 4      rlz 5      rlz 6
     0.00   0.000E+00  0.000E+00  0.000E+00  0.000E+00  0.000E+00  0.000E+00
    60.00   0.000E+00  0.000E+00  0.000E+00  0.000E+00  0.000E+00  0.000E+00
   120.00   0.000E+00  0.000E+00  0.000E+00  0.000E+00  0.000E+00  0.000E+00
   180.00   0.000E+00  0.000E+00  0.000E+00  0.000E+00  0.000E+00  0.000E+00
   240.00   0.000E+00  0.000E+00  0.000E+00  0.000E+00  0.000E+00  0.000E+00
   300.00   0.000E+00  0.000E+00  0.000E+00  0.000E+00  0.000E+00  0.000E+00
   360.00   0.000E+00  0.000E+00  0.000E+00  9.277E-08  0.000E+00  2.151E-04
   420.00   0.000E+00  0.000E+00  0.000E+00  1.510E-08  7.880E-06  7.841E-02
   480.00   0.000E+00  0.000E+00  0.000E+00  3.767E-07  1.080E-08  1.377E-01
   540.00   0.000E+00  1.673E-06  0.000E+00  1.046E-08  0.000E+00  1.905E-01
   600.00   6.539E-04  1.200E-07  0.000E+00  5.466E-09  0.000E+00  3.393E-02
```

These verification outputs numbered 6 for sources 2, 3 4, and 6 are not reproduced here but they are included in the full READPUFFS output file attached to the database.

```
6. Testing READ_PUFF_TSEQS for source 5
   icell,jcell indices  598  348   are found in 20x20 block ib,jb   30   18
   Bldg Zero, nonzero, and Open Zero, < threshold, and >= threshold counts:
   Source 5 #Bldg0 & NZ      0    0   #Open0, <TH & >TH  58650   855 10895

   Spot check density values for Source 5  in cell   598  348  Bhgt    0.00
   time(s)    rlz 1      rlz 2      rlz 3      rlz 4      rlz 5      rlz 6
     0.00   0.000E+00  0.000E+00  0.000E+00  0.000E+00  0.000E+00  0.000E+00
    60.00   0.000E+00  0.000E+00  0.000E+00  0.000E+00  0.000E+00  0.000E+00
   120.00   0.000E+00  0.000E+00  0.000E+00  0.000E+00  0.000E+00  0.000E+00
   180.00   0.000E+00  0.000E+00  0.000E+00  0.000E+00  0.000E+00  0.000E+00
   240.00   9.139E-03  0.000E+00  0.000E+00  0.000E+00  0.000E+00  0.000E+00
   300.00   1.296E-02  0.000E+00  0.000E+00  0.000E+00  0.000E+00  2.572E-02
   360.00   9.014E-04  0.000E+00  0.000E+00  4.617E-05  0.000E+00  4.245E-04
   420.00   4.547E-04  0.000E+00  0.000E+00  3.356E-04  3.401E-04  5.459E-03
   480.00   4.413E-04  1.145E-03  0.000E+00  5.329E-02  1.818E-01  6.949E-04
   540.00   7.182E-04  3.597E-03  4.713E-06  8.510E-04  3.244E-02  1.641E-03
   600.00   9.054E-04  2.096E-03  8.840E-06  3.713E-03  1.002E-01  1.154E-01
 Opened Source5_rhot.csv on unit 7
```

The data for Plume 5 in the number 6 output just above illustrates another feature of multi-realization puff dynamics. The arrival time of the puff cloud at the measurement station is different for each realization, about a factor of three above. This variability is every bit as large and just as important as the density variations. Extensive studies of this arrival-time variability are also reported in the Hamburg wind tunnel results for simulations of Hamburg Germany and Oklahoma City (Patnaik, et al., 2009; Harms, et al., 2011, Schatzmann, et at., 2011).

READPUFFS also prints out a file of comma-separated values (*Source5_rhot.csv*) that contains tracer densities for each of the 16 realizations of the Source 5 release as a function of time. These 100-minute time sequences are measured 500 meters downwind of source location 5. The table printed above for Source 5 shows a small subset of this data at a few of the time slices. A number of software packages can plot such *.csv* data including Excel. Figure 7 below shows a plot of this Source 5 data prepared with KaleidaGraph. As can be seen in the figure, the density values from all 16 realizations in this *.csv* file span about 10 orders of magnitude. The black solid curve, labeled 'average' is the average of all the realization values that are above the 10^{-12} threshold at the measurement station. The grey curve labeled 'avg+rms' above the black curve shows a density that is one rms deviation, based on the 16 current values, above the average. The curve labeled 'avg/rms+' is the same fraction lower than the average. These averages are largely meaningless when only one or two realizations are above threshold, i.e. the last half of Fig. 7.

All realizations have significant densities for the first 20 or 30 minutes but only two of them, Realizations 3 and 8 have density values at the measurement station above threshold after 30 minutes. These are illustrated as the two panels on the left in Fig. 6. Building trapping, which occurs with relatively low probability in this particular case, causes this different behavior as noted earlier.

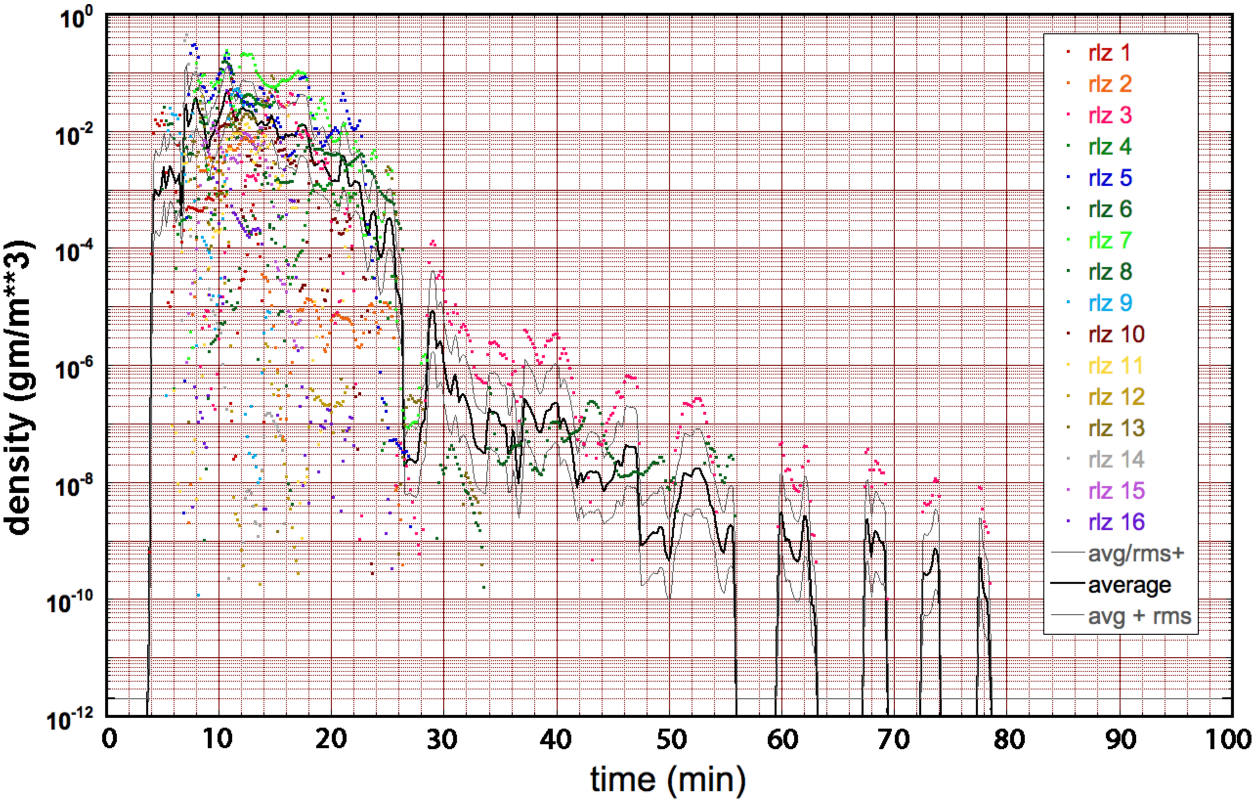

Figure 7. Density time histories for all 16 realizations of the Source 5 puff release at the indicated measurement station (purple circle) in Fig. 6.

After about forty minutes, the overall clouds have decayed sufficiently that density above the threshold is only seen as intermittent blips as the remaining plume sweeps back and forth across the measurement station. Up through about 55, minutes realizations R3 and R8 are still seen. After that only R3 has enough tracer to register as four additional blips in the next 18 minutes. A further look at the figure shows the amplitude of the R3 blips decaying exponentially over four orders of magnitude in density. This result is entirely consistent with a fixed fraction of the remaining tracer being shed during roughly periodic vortex shedding from the small building complex that is a couple of hundred meters south and west of the S5.

6. Typical Plume Examples and Results

The first impressions most people take away from viewing a movie of these evolving plumes is the pervasive fluctuations in the density and the meandering of the plume edges back and forth across the urban landscape. These effects are also observed in the wind tunnel tests and the field trials (e.g., Patnaik, et al., 2009; Harms, et al., 2011). Figure 8 below shows that time-averages of density at a point taken over a long period, such as one hour shown here, will still vary significantly from the "true" long-term average.

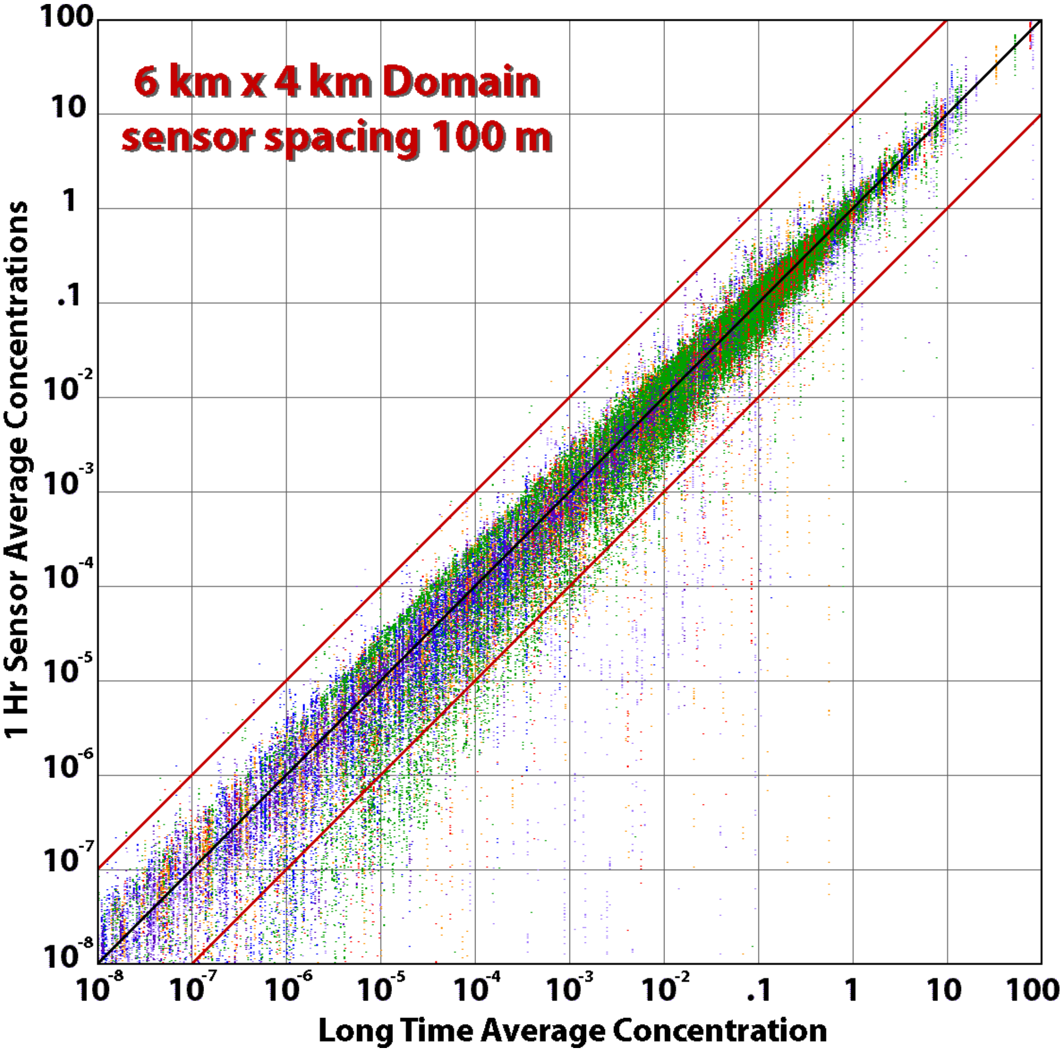

Figure 8. Scatterplot to illustrate the deviations of 1-hour average concentrations (vertical axis) from the long-term average (horizontal axis) for the 6 separate plumes in the *PLUMES_xsecs* dataset. The wind is from 300^0 at 3.0 m/s and the temperature is 0^{0}C. The data points representing the distinct plumes are plotted in different colors.

A grid of 'sensors' spaced 100 meters apart over the entire domain recorded the ground-level density of the plumes. Considering only the computational sensors outside of buildings, the plume density was averaged at these locations over the last 10,000 files (~ 6.9 hr) in the *PLUMES_xsecs* dataset for each of the six sources. There is one such long-time average at every for each source and sensor grid point. This average would become an even better approximation to the "true" average if the total time were increased. Each of these computational 'sensors' also can have a number of shorter-term averages formed from distinct portions of the entire time history. Plotting each 1-hour average on the vertical axis against the corresponding long-time average on the horizontal axis results in Fig. 8. Large average density values in the upper right of Fig. 8 correspond to the relatively few sensor locations close to a source. The points toward the lower left, corresponding to very small density averages further from each source, are somewhat more diffuse. The deviations of the 1-hour average from the long time average, particularly on the low-density side, become more extreme. As a rule of thumb, you can expect a significant number of 1-hour averages to be three or more times larger than the "true" average and an even larger number to be ten to one hundred times smaller.

The tracer density time histories in the *PLUMES_tseqs* dataset can be inspected to see how the density actually varies at representative locations. Figs. 5, 8, and Fig. 9 illustrate three physical aspects of interest. Figure 8 addressed the local variability expected of even long-time averages. Fig. 5, reinforced by the information in Fig. 7 for puffs, highlights the issue of intermittency in passive scalar transport. Figure 9 presents the Fourier transform of the six plume density traces at CFD cell 546, 246 taken over a 10,000-slice (25,000s) interval. Some of the spectra seem to be continuous curves; others seem to be composed of two straight lines.

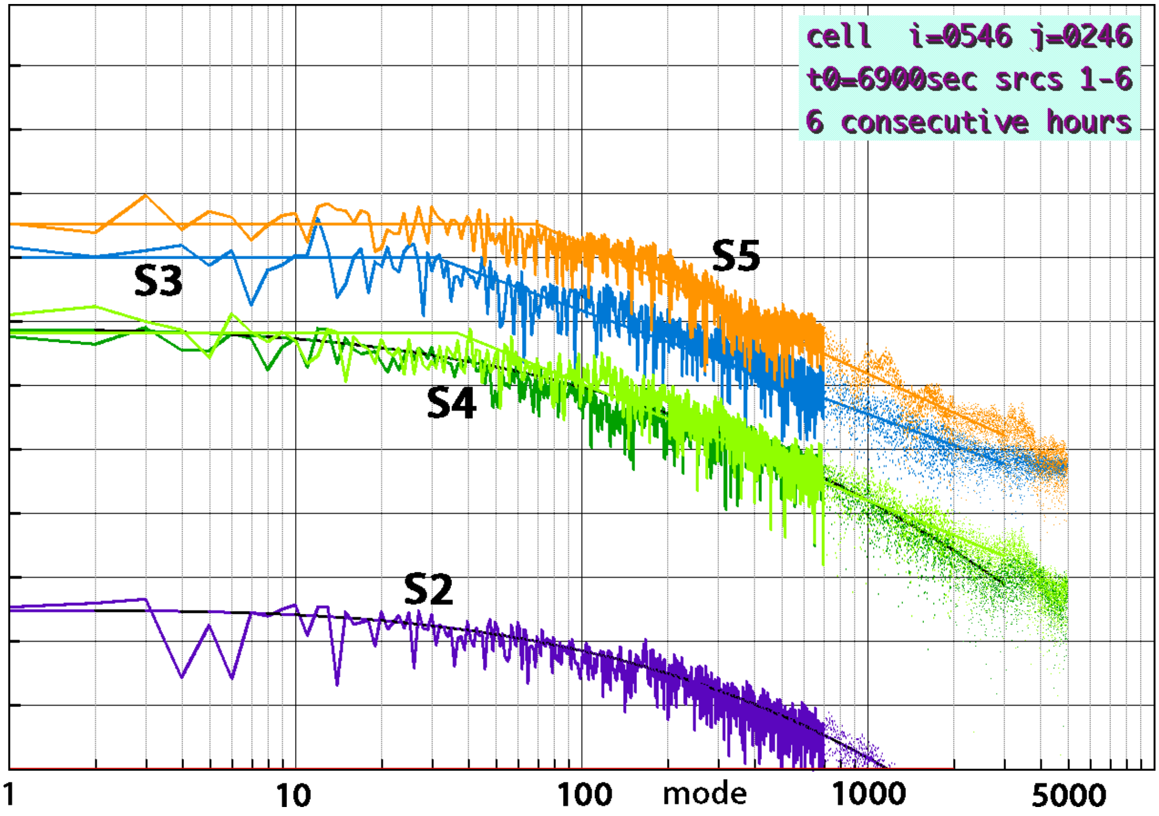

Figure 9. Fourier spectra for five of the six plumes at CFD cell 546, 246. Some of the spectra seem to be flat out to about mode 100 and then drop off with a exponential tail. Others (dark green and purple) appear more curved. Plume 6 never reaches this sensor station. Each division on the vertical axis is one order of magnitude on a logarithmic density scale.

These density time-average sequences are variable, rather than uncertain, in the sense that it should be possible to know the statistical distribution from which the "random" density values are being drawn. So what is this distribution? Figure 10 accumulates all the instantaneous density values at the grid of sensor locations from all six plumes. The vertical axis is the long-time local average density and the horizontal axis is the instantaneous density divided by the average to collapse the distributions from all locations in the plume. Both linear (panel b) and lognormal assumptions (panel a) are plotted for comparison. It has been speculated that the distribution is normal (Gaussian) with a delta function at zero density, but lognormal looks better.

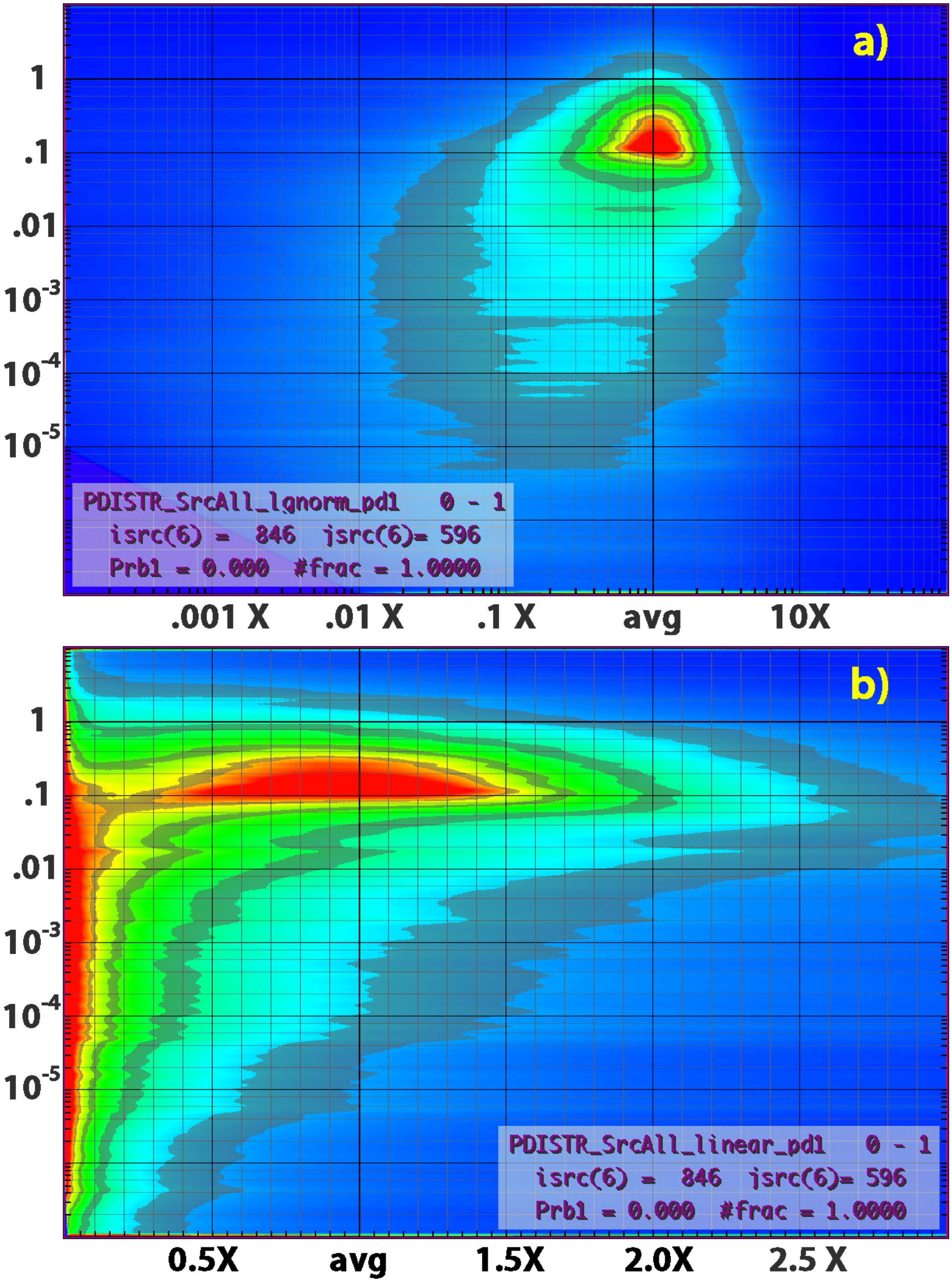

Figure 10. Instantaneous density distributions plotted on logarithmic (a) and linear (b) relative density scales. The vertical axis is the average density. The horizontal axis is the instantaneous density at each point and time divided by the long-time average density at that location. A log-normal description (panel a) seems to better fit the data.

The two-dimensional distribution of density values in Fig. 10 has data from all sources and ground-level locations superimposed in one composite distribution. The data in (a) above are collapsed into separate one-dimensional distributions in Fig. 11, one for each plume. The seventh distribution (black curve labeled ‘all 6’) is for the summed distribution of all sources. The probability distributions in Fig. 11 are for a small region near cell 1101, 701. The horizontal axis is the ratio of the instantaneous local plume density, sampled many times, to the long-time average. This normalizes each source and the different sampling locations. Because the density is non-negative, there must be many density values below average to make up for one that is 10 times larger. We can see that the low-density tails of the distributions in Fig. 11 are enhanced and thus ‘lognormal’ also does not completely characterize the ‘experimental’ distribution either.

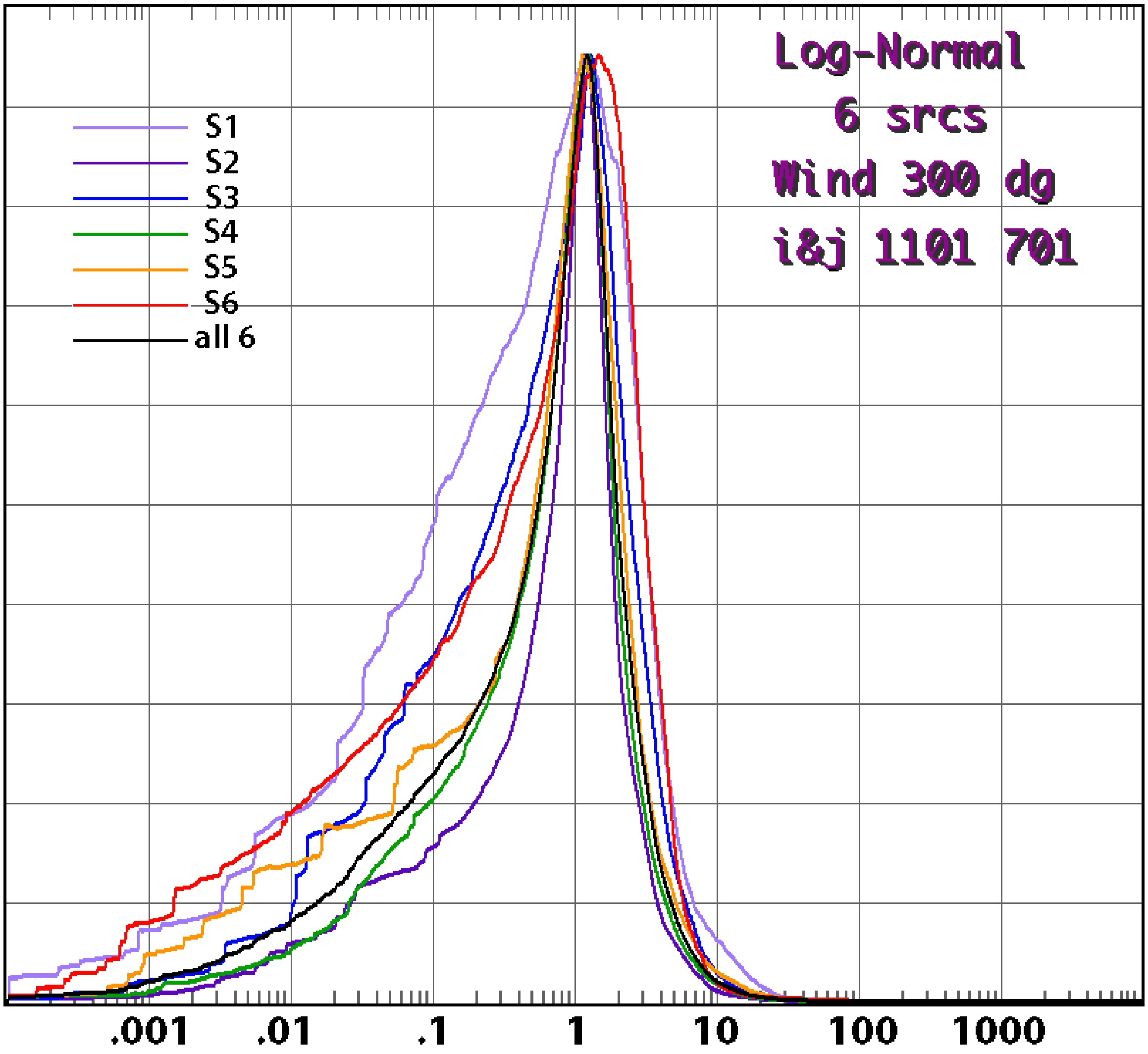

Figure 11. Distribution of density values (arbitrary units) for each source collapsed onto a single scale. The distributions are peaked near the average density (1 on the logarithmic horizontal). The distribution is near log-normal but there is a pronounced tail on the low density side approaching zero density.

These plume and puff datasets are also being used to look at the effects of fluctuations and natural variability on toxicity predictions. This test case (Boris and Patnaik, 2012, 2014) is for chlorine whose toxicity has been widely studied and is well characterized by the U.S. EPA (2019) through their Acute Emergency Guideline Levels (AEGLs). Exceeding the AEGL 1 chlorine density level marks the threshold of annoying symptoms such as watering eyes or breathing discomfort in a general population. AEGL 2 indicates the onset of dangerous, possible irreversible effects, and level 3 marks the onset of lethality.

On the left in Fig. 12 below we show an instantaneous color contour plot of the ground-level chlorine density computed by FAST3D-CT 2 hours after the continuous source S4 began emitting. The source location is marked with the yellow circle at the upper left end of the lavender high-density region, which is just to the right of the label "Source 4". The building cross-sections at ground level appear white in the figures (assuming that no chlorine penetrates). Every 2.5 seconds FAST3D-CT saves the computed ground-level density and sums this cross-section to compute the running density average. This average is shown after 2 hours on the right in Fig. 4. The time-averaged densities are smoother and appear to extend beyond the bounds of the instantaneous plume in some locations. This accurate running average approximates what steady-state/time-averaging/ensemble models might predict the density to be. The density differences between instantaneous and time-averaged densities do not appear to be great when viewed with a logarithmic color map as in Fig. 12 below, but their effect on the AEGL hazard areas, computed using the two density fields, is important.

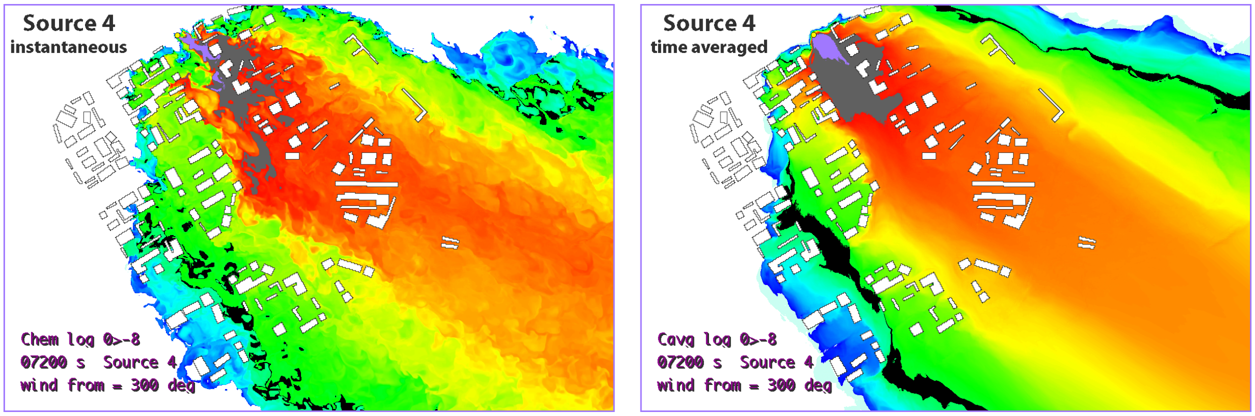

Figure 12. Instantaneous plume density (left) and time averaged plume density (right) for continuous release of chlorine from Source 4 located in the upper left of a 6 km by 4 km domain. Density is shown on a logarithmic scale with black bands indicating concentration values near 0.05 ppm.

Figure 13 below compares the AEGL exposure hazard area predictions, based on the FAST3D-CT time-varying plume, using the new time-dependent AEGL integration routines in the EAGLE package (Boris and Patnaik, 2012, 2014). The two panels below show the AEGL 1 (yellow), AEGL 2 (red), and AEGL 3 (black) hazard areas with and without the natural density fluctuations that are computed in 3D detail by the FAST3D-CT simulation of the fluctuating and gusting winds through the urban geometry. The figure also lists the predicted hazard areas in square kilometers for the two contrasting cases. The area ratios are written on the right hand panel for comparison.

Including realistic agent density fluctuations increases the hazard area even when the other conditions are all the same. This is expected as it has been recognized for some time that toxic load accumulates nonlinearly in higher-density regions for many agents. This is further corroborated in Fig. 13 by the fact that the AEGL 1 area ratio, based on a limiting threshold of 0.5 ppm, is larger than the AEGL 2 and AEGL 3 area ratios that result from a nearly quadratic dependence of accumulation rate with concentration. Interpreting a threshold behavior as a power law requires the exponent used to be infinity since the onset time varies discontinuously with almost no change in the density. These fluctuation-augmented hazard areas in Figs. 12 and 13 are also typical of the other five plumes calculated.

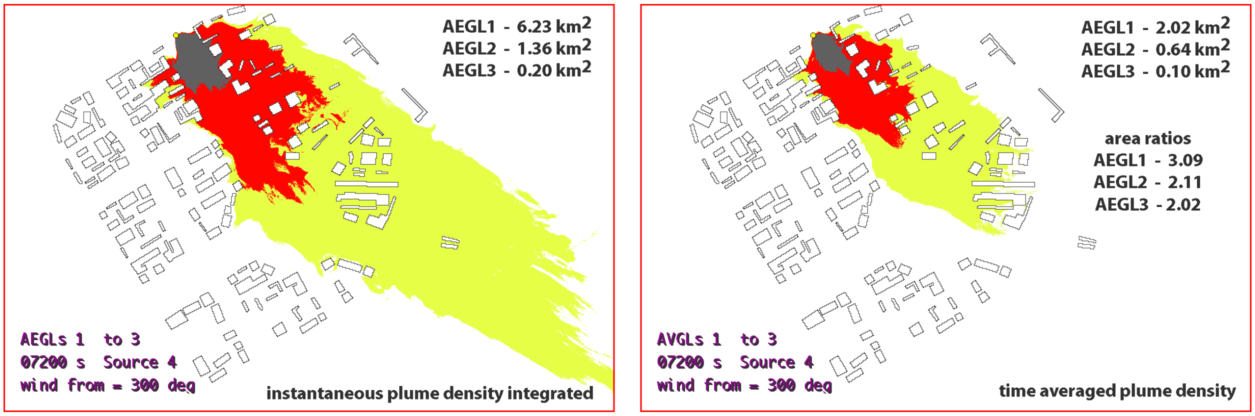

Figure 13. AEGL 1 (yellow), AEGL 2 (red), and AEGL 3 (gray) exposure hazard areas at 2 hours after release computed using the new generalized AEGL routines to integrate the actual, ground level, time-varying densities (left) and the corresponding time-averaged density profiles (right). Natural fluctuations in the density (left), interacting with the non-linear toxic-load behavior of the EPA AEGL tables, increase the hazard areas appreciably.

Area ratios in excess of 2 or 3 are sometimes seen, as estimated by Bogen and Gouveia (2008). When the computed exposure hazard areas extends beyond the computational domain, as happen with a stronger source in Fig. 13, the hazard area cannot be computed everywhere and the area computed is too small. Since the hazard area computed with fluctuations extends beyond the FAST3D-CT simulation domain before the averaged-density hazard area does, the correct, overall area ratios are still larger than we are able to compute and have shown above.

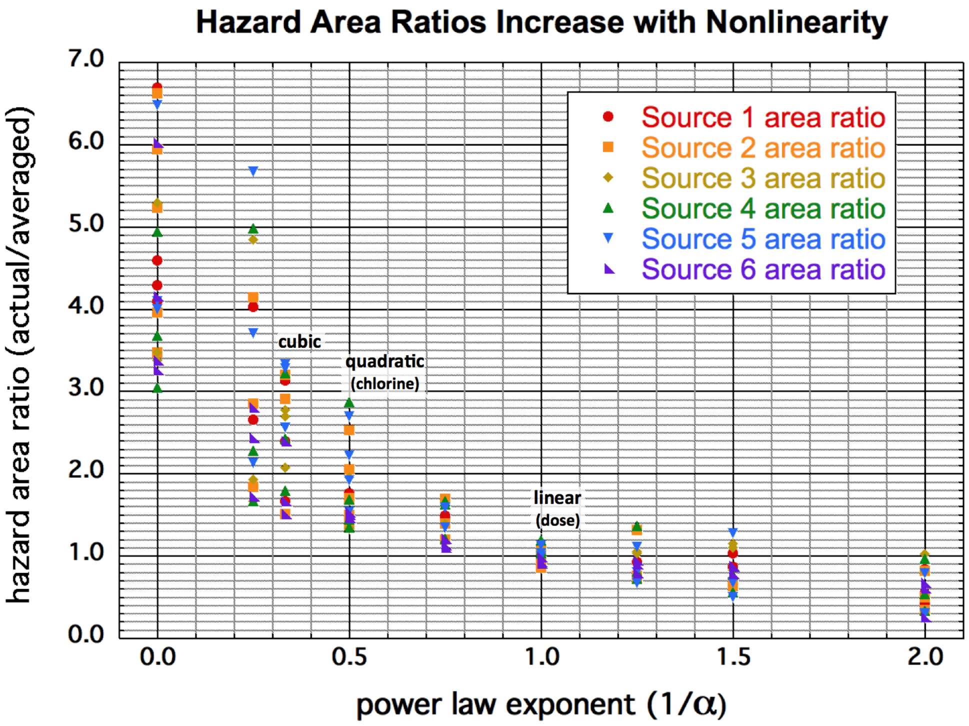

Figure 14. AEGL exposure area ratios for all six sources and a range of power-law exponents. The threshold hazard area behavior, shown at 1/a = 0 near the left edge of the figure, may even extend above an area ratio of 7:1 in some cases.

Figure 14 above summarizes the results of these exposure hazard-area ratios as a function of the toxic load power law exponent. As the nonlinearity of the toxicity, and thus the power-law exponent, we move to the left in Fig. 14. At 0.0, which is an idealized threshold behavior, the hazard areas may be increased three to seven times over those found using time- or space-smoothed average densities.

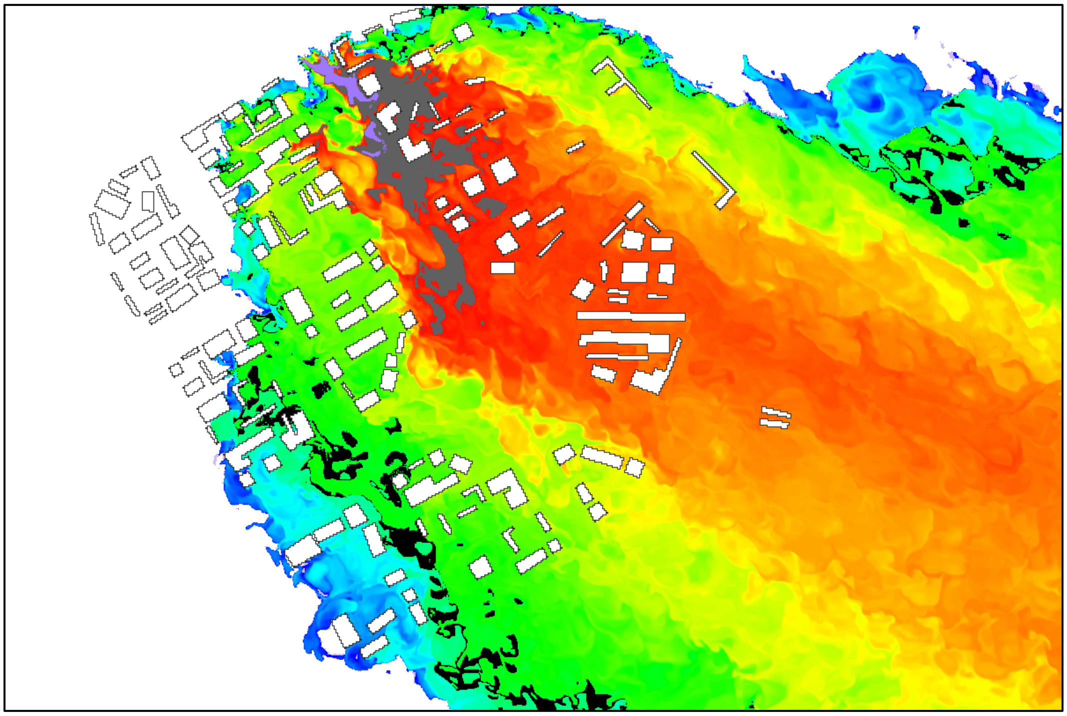

Figure 15. Plume density for Source 4 at ground level 2 hours after start of tracer release (step 87,000), rebuilt from the 2400 blocks of time sequence data. The threshold hazard area behavior, shown at 1/a = 0 near the left edge of the figure, may even extend above an area ratio of 7:1 in some cases.

Figure 15 shows the density cross-section at 2 hours after the beginning of the continuous release S4. This plot was prepared recently from an adaptation of READPLUMES using the *PLUMES_tseqs* dataset with a local plot package added. This cross-section was reconstructed from the time sequences by reading each of the 2400 time-sequence blocks and assembling all the 20x20 squares of values for the single time step 87,000 into a single array. This plot is identical to the leftmost panel of Fig. 12, prepared several years ago. This test was performed to demonstrate the absolute consistency of the cross-section and time sequence data sets and the lossless nature of the 'lz4' data compression.

7. Typical Puff Examples and Results

Many of the analyses illustrated above for tracer plumes can be performed for tracer puff clouds as well - with the added complication that even the ensemble-averaged solutions will be time dependent. Since tracer species blow away with a decay time scale of a half hour or so, the true long-time average is always zero. This means some other way must be found to normalize distributions. As noted earlier, naturally occurring fluctuations cause identical releases from any particular location, which differ only in their release time, to differ greatly. This variability is captured in datasets *PUFF_xsecs* and *PUFF_tseqs* by including 16 separate realizations, R1 through R16, of each instantaneous release. These different realizations were initialized every 60,000 timesteps and this are totally separate in time. Each independent realization lasts for 1 hour 40 minutes.

This variability for an ensemble of realizations of the puff release S2 is shown in Fig. 16. This is an instantaneous snapshot combining 8 of the 16 realizations as measured in one cell. In each realization, the tracer arrives at different times but the decay times all appear to be very similar.

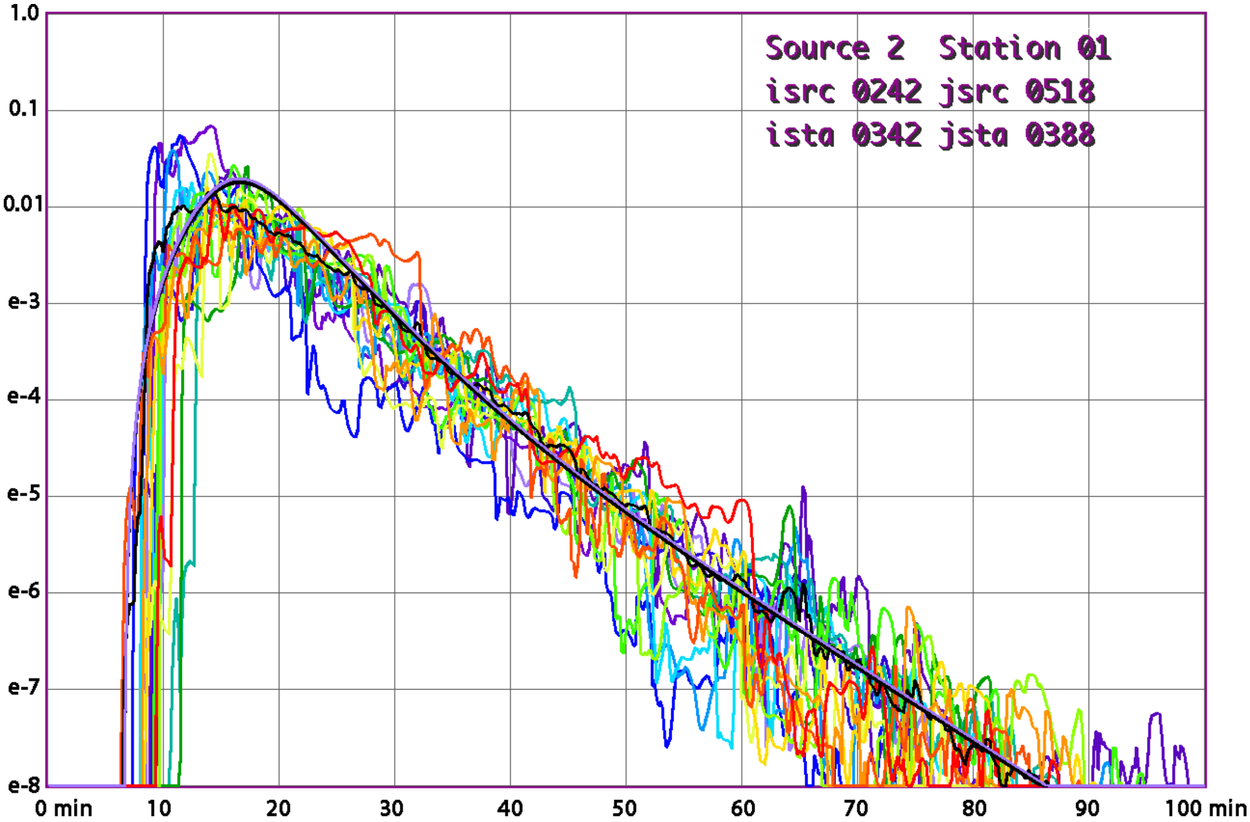

Figure 16. Density time sequences for 8 of the 16 realizations of S2 measured at CFD cell 342, 388. The tracer arrival time for this sensor location varies by almost a factor of two. The densities also fluctuate by more than an order of magnitude among themselves over the duration of the experiment. The black curve represents the average density time history and the superimposed smooth lavender and black curve fits an analytic function approximating the average time behavior.

Figure 17 shows the probability that Source 5 will contaminate each point in the domain at three well-separated times for the ensemble of 16 realizations of S5. A threshold density of 10^{-8} gm/m^3 was used below which the density was taken as zero for counting realizations. The lavender area is the area where all 16 realizations overlap while the lightest blue area, primarily around the periphery, marks where only one realization (6.25% probability) can be found. This display shows clearly the strong influence of the buildings on the probability of contamination.

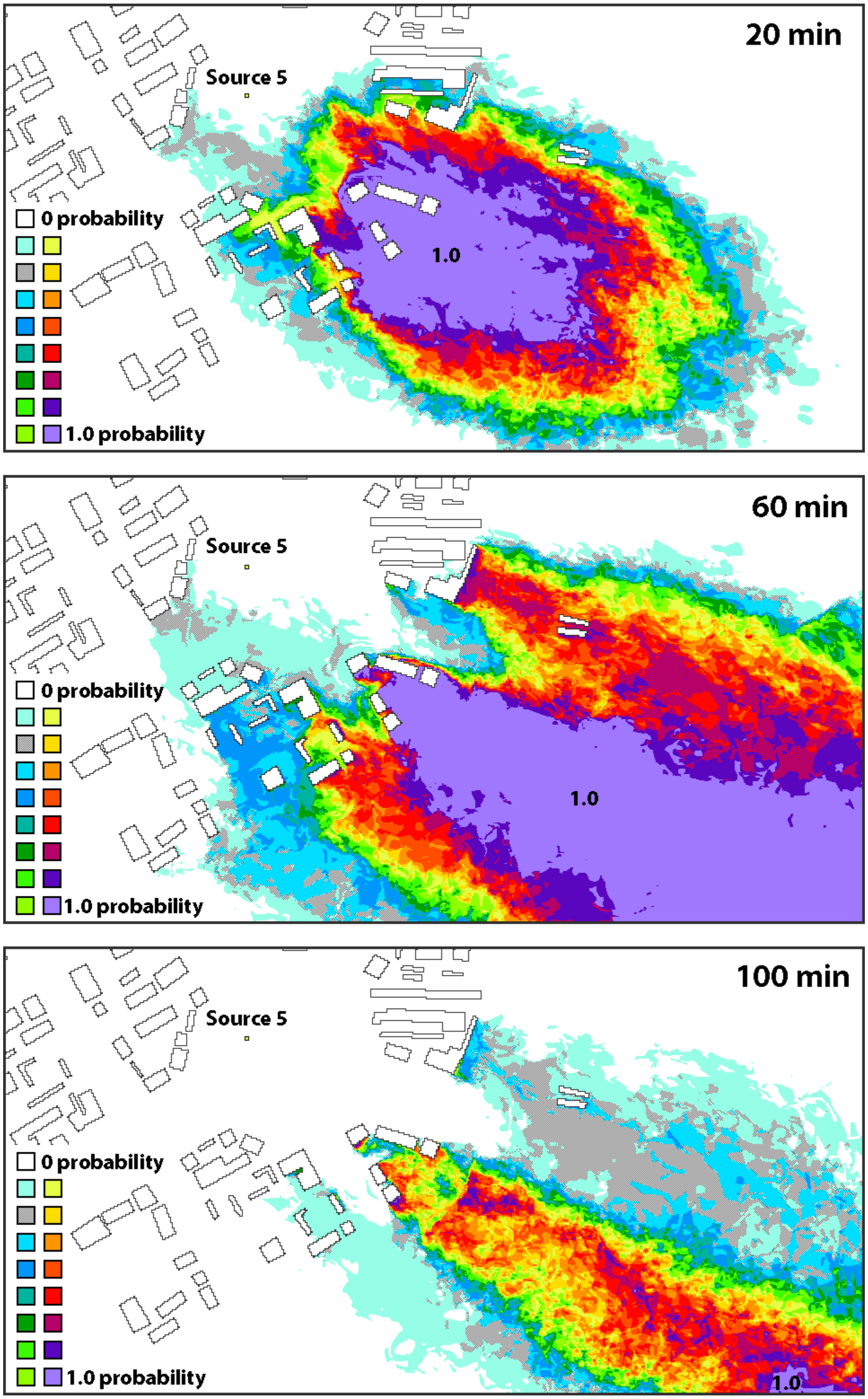

Figure 17. Probability of contamination above a threshold of 10^{-8} gm/m^3 after 20, 60, and 100 minutes for Plume 5. Only a 4.8 km by 2.5 km portion of the full region is shown. Only one and two of the 16 realizations reach the pale blue and grey regions respectively. The lavender central region marks where all 16 realizations are present above threshold at the indicated time.

8. Summary

Four compressed datasets, extracted from two extensive FAST3D-CT Monotone Integrated Large-Eddy Simulations (MILES), are described and illustrated in this report. These datasets were designed to provide data on passive, neutral-buoyancy, gas-tracer Contaminant Transport (CT) from a very well validated and detailed simulation model. The scope and detail of the datasets allow users to focus on the realistic, large-scale fluctuations that occur in urban airflow. The datasets are constructed to facilitate study of both the expanding plumes from continuous sources and the transient "puff" clouds from acute (instantaneous) sources. Sensor response can be simulated in realistic fluctuation environments (e.g., Harms, et al., 2011; Leitl, et al., 2015), accurate health effects can be computed for a number of scenarios and agents (Boris and Patnaik, 2014, 2016); and the detailed fluid dynamics of complex passive tracer transport can be evaluated, as can be shown by the examples in Sections 6 and 7. These data can also be used to validate simpler plume and puff models (Moses, et al., 2015; Boris and Patnaik, 2019; Boris, et al., 2019).

A sponsor's requirement for a few detailed CFD cases in a particular type of geometry determined the nature of the buildings and terrain. Many parameters had to be fixed for these datasets. The extent of these time-dependent runs and the amount of data needed to approach statistical relevance for each prevented including different inflow profiles and wind directions in the database. For years we have used one-ton releases for the puffs as a maximum plausible source for some chemicals but the choice of 100 kg/sec for the continuous sources may seem excessive. These are passive tracers, however, so a user can scale these densities to whatever makes sense for his application. Figure 18 shows the velocity and temperature profiles actually used. 3 m/s at 100-meter altitude is a reasonable, representative speed for light wind conditions and the temperature rise of ~1 degree every 200 meters is a shallow stabilizing gradient. This gradient has the effect of leaving the tracer species in a nearly neutrally buoyant state, particularly within the urban canopy under 100-meters where turbulent vertical transport can be expected to disperse the tracer when the stratification is weak. Since the viscosity appears nowhere in the computations, users can adjust the space and time scales for other applications.

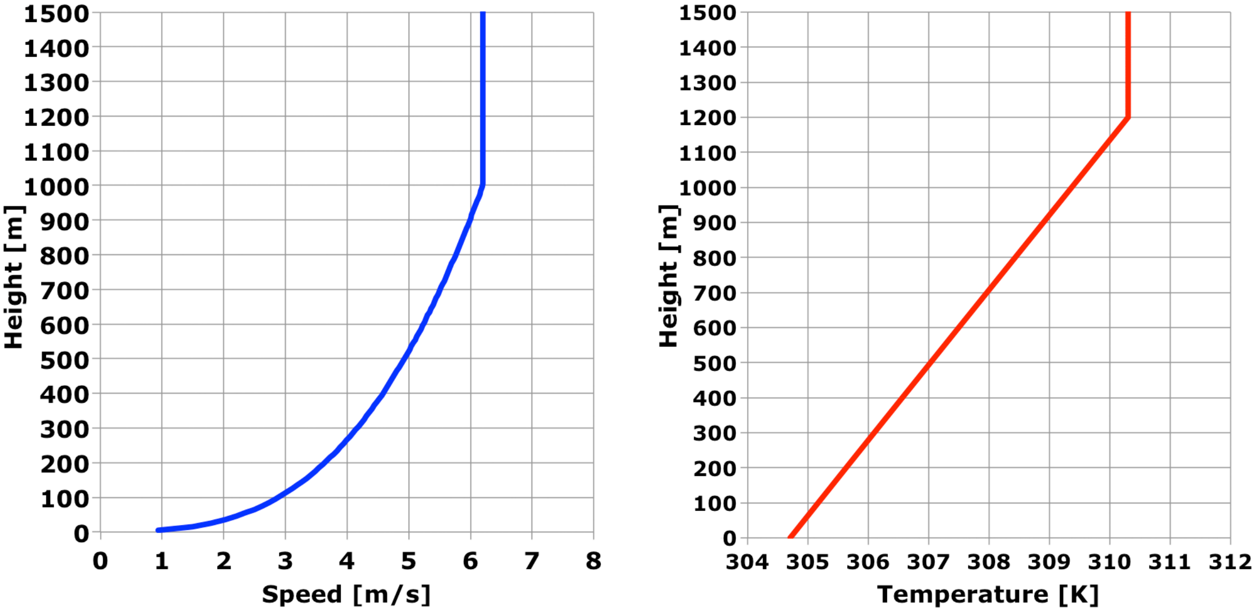

Figure 18. Idealized inflow and boundary condition profiles for the "Puffs and Plumes" database. A shallow but stabilizing temperature gradient was chosen to force the tracer to stay near the ground without rapid vertical dissipation.

Two short test programs are included as Appendices B and C to illustrate the use of the simple Fortran utilities included as Appendix C. These utilities are provided to read data from the four compressed datasets. The total storage required, less than 253 GB when compressed using 'lz4', was chosen to fit conveniently on a single, slim-format, USB 3.0 pocket drive. Rather than try to make this data available on the web for down loading with rather long transmission times, we recommend copying the data onto a clean pocket drive provided by any potential user and send the drive, with data and programs, back to the user.

In closing, the examples discussed in Sections 6 and 7, the two "Typical Examples and Results" sections, and the content of the test programs were chosen to highlight the importance of naturally occurring fluctuations to the physics and turbulent fluid dynamics in a urban geometry. In turn users may wish to study their importance to the potential human consequences of exposure to agents that may simply be pollutants or may be something worse. We hope other scientists will find this data useful and perhaps contribute extensive datasets of their own in the future. We are planning several future papers using these datasets to delve into the several scientific issues raised in some of the examples above.

References

Bogen, K.T. and F.J. Gouveia (2008), Impact of spatiotemporal fluctuations in airborne chemical concentration on toxic hazard assessment, *Journal of Hazardous Material*, Volume 152, Issue 1, 21 March 2008, Pages 228–240.

Boris, J.P. and G. Patnaik, 2019, Proximity Analysis for Model Calibration and Validation, U.S. Naval Research Laboratory (Draft NRL memo report), Jan 2019.

Boris, J.P., S. Fischer, K. Storm, G. Patnaik, D. Hertwig, B. Leitl, F. Harms, A. Moses and K. Obenschain, 2019, Modeling Atmospheric Contaminant Transport for Real-Time Response in Cities, *journal article in preparation*, Jan 2019.

Boris, J.P., G. Patnaik, and K. Obenschain, 2019, Health Effect Hazard Areas for Airborne Contaminant Agents, NRL/MR/6003-19-9843, Jan 2019.

Boris, J.P. and G. Patnaik, 2016, Neutralization of Airborne Contaminants, AIAA Paper 2016–3196. Aviation 2016, Washington DC, AIAA, 13 June 2016.

Boris, J.P. and G. Patnaik, 2014, Acute Exposure Guideline Levels (AEGLs) for Time Varying Toxic Plumes, U.S. Naval Research Laboratory Memorandum Report NRL/MR/6003-14-9493, September 2014, Washington DC.

Boris, J.P. and G. Patnaik, 2012, Acute Exposure Guideline Levels (AEGLs) for Time Varying Toxic Plumes, 16th GMU Conference on Atmospheric Transport and Dispersion, George Mason University, Fairfax VA, 17 July 2012.

Boris, J.P., G. Patnaik, and Keith Obenschain, John Delaney, and John Donnelly (2010), Fast and Accurate Prediction of Windborne Contaminant Plumes for Civil Defense in Cities, Proceedings: Computational Wind Engineering Conference 2010 (CWE'2010), 23–27 May 2010, Chapel Hill, NC.

Boris, J.P., G. Patnaik, M-Y. Lee, T. Young, B. Leitl, F. Harms and M. Schatzmann (2009), Validation of an LES Urban Aerodynamics Model for Homeland Security, 47th AIAA Aerospace Sciences Meeting, *AIAA Paper 2009–1633*, Orlando FL, 5–8 January 2009, (AIAA, Reston VA, 2009).

Boris J.P., 2005, Dust in the Wind: Challenges for Urban Aerodynamics, AIAA 2005 Fluid Dynamics Prize Plenary Lecture, 17th AIAA Fluid Dynamics Meeting, Toronto Canada, 4–8 June 2005, *AIAA Paper 2005–5393*. AIAA, Reston VA.

Boris, J.P., 2002, The Threat of Chemical and Biological Terrorism: Preparing a Response, *Computing in Science and Engineering* **4**(2), pp. 22–32.

Boris, J.P., F.F. Grinstein, E.S. Oran and R.L. Kolbe, 1992, New Insights into Large Eddy Simulation, *Journal of Fluid Dynamics Research* **10 (4-6)**, pp. 199-228, (also U.S. Naval Research Laboratory Memorandum report NRL/MR/4400-92-6979).

Boris J.P., 1989, On Large Eddy Simulation Using Subgrid Turbulence Models, in **Whither Turbulence? Turbulence at the Crossroads**: Lecture Notes in Physics; 357: 344–353, The 1989 Cornell Conference, J.L. Lumley (ed), Springer-Verlag, New York, 1990.

Boris J.P., and D.L. Book, 1976, Solution of Continuity Equations by the Method of Flux-Corrected Transport. Chapter 3, **Methods in Computational Physics 1976**; 16: 85–129. Academic Press, New York.

Boris J.P. and D.L. Book, 1973, Flux-Corrected Transport I. SHASTA, A Fluid Transport Algorithm That Works, *Journal of Computational Physics* 1973; 11(1): 38–69. {This became the most cited NRL journal article, was reprinted in full in the 25th anniversary edition of *JCP*, 135(2): 172–186, August 1997, and the subject of “Flux-Corrected Transport Looks at Forty” *Computers & Fluids* (2013), j.compfluid.2013.05.004, http://dx.doi.org/10.1016/, Elsevier}.

Boris J.P., 1971, A Fluid Transport Algorithm That Works. Computing as a Language of Physics. IAEA–SMR 1971; 9(18): 171–189. Presented at Centre for Theoretical Physics 1970, Trieste Italy, U.S. Naval Research Laboratory Memorandum Report 2357, 1971.

Britter, R.E. and Hanna, S.R., 2003, “Flow and Dispersion in Urban Areas,” *Ann. Rev. Fluid Mech.* **35**, 2003, pp. 469-496.

Fischer, S., P. Rechenbach, K. Storm, M. Schatzmann, B. Leitl, D. Hertwig, F. Harms, G. Patnaik, and J.P. Boris, 2012, “Mit CT-Analyst in Storfalleinsatz vor die Lage Kommen,” *Bevolkerungsschutz of the German Federal Office of Civil Protection and Disaster Assistance*, April 2012.

Grinstein F.F., L.G. Margolin, W.J. Rider (eds), 2010, **Implicit Large Eddy Simulation: Computing Turbulent Fluid Dynamics**, Cambridge University Press. 2007; 2nd printing 2010.

Harms, F., B. Leitl, M. Schatzmann, and G. Patnaik, 2011, Validating LES-based Flow and Dispersion Models, *Journal of Wind Engineering and Industrial Aerodynamics*, Vol. 99, pp 289-295, April 2011.

Harms, F., B. Leitl, M. Schatzmann, G. Patnaik, M-Y. Lee, T.R. Young, and J.P. Boris, 2007, Statistical Representativeness of Puff Dispersion Data Measured in a Complex Urban Environment, Proceedings: 11th Annual George Mason University Conference on Atmospheric Transport and Dispersion Modeling, Fairfax, VA, 10–12 July 2007.

Hertwig, D., G. Patnaik, B. Leitl, 2016a, LES validation of urban flow, part I: flow statistics and frequency distributions, *Environmental Fluid Mechanics,* Springer, DOI 10.1007/s10652-016-9507-7, December 2016.

Hertwig, D., G. Patnaik, B. Leitl, 2016b, LES validation of urban flow, part II: eddy statistics and flow structures, *Environmental Fluid Mechanics,* Springer, DOI 10.1007/s10652-016-9504-x December 2016.

Holt, T., J. Cummings, C. Bishop, J. Doyle, X. Hong, S. Chen, and Y. Jin, 2011, Development and testing of a Coupled Ocean-Atmosphere Mesoscale ensemble Prediction System COAMPS-OS). *Ocean Dynamics*, **61**, 1937-1954.

Holt, T., J. Pullen, and C. H. Bishop, 2009, Urban and ocean ensembles for improved meteorological and dispersion modeling of the coastal zone. *Tellus*, **61A**, 232-249.

Kuzmin D., R. Löhner, S. Turek (eds), 2005, *Flux-Corrected Transport: Principles, Algorithms, and Applications,* Springer-Verlag Berlin, 2005; 2nd edition 2012.

Leitl, B., F. Harms, E. Berbekar, J. Boris, G. Patnaik, K. Obenschain, S. Fischer, 2016, Local-Scale Hazmat Dispersion Modeling for First Responders Based on High-Resolution Computational Fluid Dynamics - an Overview of CT-Analyst Hamburg, *Chemical Engineering Transactions,* Vol 48, 2016, ISBN 978-88-95608-39-6, June 2016.

Leitl, B., F. Harms, E. Berbekar, G.Patnaik, J.P. Boris, M. Schatzmann, 2015, Wind Tunnel Experiments or Advanced CFD ... What Do We Need for Understanding Flow and Dispersion in The Lower Atmospheric Boundary Layer?, International Conference on Modeling Fluid Flows CMFF'15, Budapest, Hungary, 1-4 September 2015.

LeitL, B., D. Hertwig, F. Harms, M. Schatzmann, G. Patnaik, J.P. Boris, K. Obenschain, S. Fischer and P. Rechenbach, 2013, Large Eddy Simulation of Accidental Releases, in A. Talamelli, M. Oberlack and J. Peinke (Eds.), *Progress in Turbulence V, Springer Proceedings in Physics* **149,** 2013.

Moses, A., K. Obenschain, J.P. Boris, and G. Patnaik, 2015, Using Real-Time Chemical Plume Models in Virtual Training Systems, IEEE International Symposium on Technologies for Homeland Security, Waltham, MA, April 14-16, 2015.

Oberkampf, W. and M. Barone, 2004, "Measures of Agreement Between Computation and Experiment: Validation Metrics," AIAA Paper, AIAA-2004-2626, 34th AIAA Fluid Dynamics Conference and Exhibit, Portland, Oregon, June 28-1, 2004.

Oberkampf, W. and J. Helton, 2002, "Investigation of Evidence Theory for Engineering Applications," AIAA Paper AIAA-2002-1569, 43rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, Denver, Colorado, Apr. 22-25, 2002

Patnaik, G., J.P. Boris, M-Y Lee, T.R. Young, Jr., B. Leitl, F. Harms, and M. Schatzmann, 2009, Validation of an LES urban aerodynamics model with model and application specific wind tunnel data, Proceedings: 7th Asia-Pacific Conference, Taipei, Taiwan, 8–12 Nov 2009.

Patnaik, G, F.F. Grinstein, J.P. Boris, T.R. Young,Jr, and O. Parmhed, 2007, Large Scale Urban Simulations, Chap. 17, pp 502–530, in **Implicit Large Eddy Simulation: Computing Turbulent Flows**, F. Grinstein, L. Margolin, and W. Rider (eds), Cambridge Univ. Press, 2007.

Patnaik, G. and J.P. Boris, 2005, Fast and Accurate CBR Defense: Bringing HPC to the Warfighter and First Responder, AIAA Paper 2005-7166, 1st AIAA InfoTech@Aerospace Meeting, Arlington VA, 26–29 September 2005.

Patnaik, G., J.P. Boris, F.F. Grinstein, J.P. Iselin, and D. Hertwig, 2005, Large scale urban simulations with FCT, pp. 91–117, in **Flux-Corrected Transport: Principles, Algorithms, and Applications**, D. Kuzmin, R. Lohner, and E. Turek (eds), Springer-Verlag, Berlin, 2nd edition, 2012, also 1st edition, pp. 105–130, 2005.

Sagaut, P., 2004, **Large Eddy Simulation for Incompressible Flows**, 2nd Edition, Springer.

Schatzmann, M., B. Leitl, D. Hertwig, F. Harms, C. Peeck, G. Patnaik, J.P. Boris, K. Obenschain, S. Fischer and P. Rechenbach, 2011, An LES-Based Microscale Airborne Hazard Model, Proceedings: 14th International Conference on Harmonization within Atmospheric Dispersion Modeling for Regulatory Purposes, Kos Island, Greece, 2–6 October 2011.

U.S. Environmental Protection Agency AEGL Program (2019), https://www.epa.gov/aegl.

Young, T.R. Jr., A.M. Landsberg and J.P. Boris, 1993, "Implementation of the Full 3D FAST3D (FCT) Code Including Complex Geometry on the Intel iPSC/860 Parallel Computer," *High Performance Computing 1993 – Grand Challenges in Computer Simulation*, A. Tentner (ed), 1993 SCS Simulation Multiconference, Arlington VA, Mar/Apr 1993, SCS, San Diego.

Acknowledgements

The authors wish to thank Gopal Patnaik, Lisa Williams, Terri Faison, Adam Moses, David Fyfe, Theodore Young, Jr. and Teddy Holt, the NRL contaminant transport team who have made these computations possible in many ways. ONR, through NRL's Core Research Program, has supported this database effort and the continuing related research. The ONR Naval Expeditionary Dog Program, Hamburg Germany, through the University of Hamburg, DHS, and the DoD High Performance Computing Modernization Program have, in the past, all contributed to aspects of the software technology used and discussed here.

Appendix A: Subroutines for Reading Plumes and Puffs from the Database

```
c  ******************************************************************.*********

      Subroutine READ_PUFF_XSECS ( xsecs, istep, kr, NX, NY, Nsrc )

c  __________________________________________________________________.__________

c  'READ_PUFF_xsecs' reads in the PUFF XY density cross-sections for all sources
c  for the single realization 'kr' (kr = 1 to Nrlz).  The results are put into
c  the real cross-section array 'xsecs.'  Density values in cells blocked out
c  by the building geometry have been set to hard zero in this dataset.  The
c  main function of this routine is to construct the correct file name and path
c  to retreive and decompress the set of Nsrc cross-sections identified by the
c  arguments.

c  xsecs    Real*4 array dimensioned NX x NY x Nsrc, i.e. 6 ground-level planes
c  istep    Timestep selected.  istep must be multiple of 25 for realization 1
c           (kr = 1) and a multiple 100 for the remaining 15 realizations.
c  kr       Realization to be read,  kr may be 1 through Nrlz = 16.
c  NX       X extent of grid,  used for dimensioning xsecs   (here 1200)
c  NY       Y extent of grid,  used for dimensioning xsecs   (here 800)
c  Nsrc     Number of sources, used for dimensioning xsecs   (here 6)

c  Program written initially  Jay P. Boris  21 Jul 2013
c  Most recent modifications  Jay P. Boris  12 Dec 2016
c  __________________________________________________________________.__________
c  Declare the main program data and associated local variables. . .
c  __________________________________________________________________.__________
      Implicit  NONE
      Integer,intent(in)::  istep, kr, NX, NY, Nsrc
      Real,intent(out)::    xsecs(NX,NY,Nsrc)    ! Density values at 'istep'

      Logical,save::        FirstCall = .true.
      Integer               i, kstep, recl, JSTAT
      Character*80          LOCATION, XsecFile, FilePath
      Character*6,save::    numb6(0:999999)
c  __________________________________________________________________.__________

c  Compute once and save a set of 6-digit character strings for naming files ...
c  __________________________________________________________________.__________
      If ( FirstCall ) Then
         FirstCall = .false.
         Do i = 0, 999999
            Write ( numb6(i), '(I6.6)' ) i
         End Do
      End If

c  Now read the density cross-section for realization 'kr' for all Nsrc sources.
c  __________________________________________________________________.__________
      kstep = istep + 60000*(kr - 1)    ! Each realization begins @ later steps
      If ( kr.lt.1 .or. kr.gt.16 ) Then
         Write ( *,* ) 'READ_PUFF_XSECS error: kr out of range ', kr
         Stop
      End If
      If ( kr .eq. 1 ) Then
         If ( mod(istep,25) .ne. 0 ) Then
            Write (*,*) 'READ_PUFF_XSECS error: kr, istep', kr, istep
            Stop
```

```
            End if
         Else If ( mod(istep,100).ne.0 ) Then
            Write (*,*) 'READ_PUFF_XSECS: istep error', istep
            Stop
         End If

         LOCATION = './PUFFS_xsecs/PUFFS_rlznn/'
         XsecFile = 'puffs_rlznn_'          ! Folder and cross-section file name
         LOCATION(24:25) = numb6(kr)(5:6)   ! Realization file name in LOCATION
         XsecFile(10:11) = numb6(kr)(5:6)   ! Realization ;kr; file name in folder
         FilePath = trim(LOCATION)//trim(XsecFile)//numb6(kstep)//'.lz'
         recl = NX*NY*Nsrc*4                ! Nsrc density cross-sections in bytes
         Call lz4_read( 20+kr, trim(FilePath), xsecs, recl, JSTAT )
         Close (unit=20+kr)   ! Multiple different unit names for parallelization.

         Return
         End Subroutine READ_PUFF_XSECS

c  *******************************************************************.*********

         Subroutine READ_PUFF_TSEQS ( tseqs, ks, ib, jb, Ntms, Nrlz )

c  ___________________________________________________________________.__________

c  'READ_PUFF_TSEQS' reads in a single 20-cell by 20-cell contiguous block of
c  puff density time sequences for source 'ks.'  All Nrlz realizations have been
c  collected into a single file.  The results are put real array 'tseqs.'
c  Density values in cells blocked out by the building geometry have been set to
c  hard zero in this dataset.  The main function of this routine is to construct
c  the correct file name and path to retreive and decompress the block of Nrlz
c  time-sequences identified in the argument list.

c  tseqs    Real*4 array dimensioned 20 x 20 x Ntms x Nrlz contains density time
c           sequences for 400 contiguous ground-level CFD cells located in block
c           ib, jb of the 2400 blocks covering the 1200 x 800 domain spanning
c           6 km by 4 km.
c  ks       Source to be read, here ks = 1, ..., Nsrc = 6.  Since sources are in
c           different locations, different blocks(ib,jb) will have to be chosen
c           to obtain the same position relative to the different sources.  Thus
c           combining realizations makes sense for time sequences but combining
c           sources does not.
c  ib       Horizontal block index:  ib = (icell+19)/20  (ib = 1, ..., 60)
c  jb       Vertical block index:    jb = (jcell+19)/20  (jb = 1, ..., 40)
c           where icell and jcell are the indices of the CFD cell of interest.
c           Block ib,jb contains cells   icell = 1 + 20*(ib-1) through 20*ib
c           with jcell = 1 + 20*(jb-1) through 20*jb.
c  Ntms     Number of slices in entire time sequence (here 601, rlz 1 decimated)
c  NRlz     Number of puff realizations, used for dimensioning xsecs  (here 16)

c  Program written initially  Jay P. Boris  23 Jul 2013
c  Most recent modifications  Jay P. Boris  12 Dec 2016
c  ___________________________________________________________________.__________
c  Declare the main program data and associated local variables. . .
c  ___________________________________________________________________.__________
         Implicit  NONE

         Integer,intent(in)::  ks, ib, jb, Ntms, Nrlz
         Real,intent(out)::   tseqs(20,20,Ntms,Nrlz)  ! time sequences for src 'ks'
```

```
      Logical,save::       FirstCall = .true.
      Integer              i, recl, ISTAT
      Character*80::       LOCATION, TseqFile, FilePath
      Character*8::        BlockID
      Character*6,save::   numb6(0:999999)
c  ___________________________________________________________________.__________

c  Compute once and save set of 6-digit character strings for naming files ...
c  ___________________________________________________________________.__________
      If ( FirstCall ) Then
         FirstCall = .false.
         Do i = 0, 999999
            Write ( numb6(i), '(I6.6)' ) i
         End Do
      End If

c  Now read the time histories for src 'ks' for all Nrlz sources.
c  ___________________________________________________________________.__________
      If ( ks.lt.1 .or. ks.gt.6 ) Then
         Write ( *,* ) 'READ_PUFF_TSEQS error: ks out of range ', ks
         Stop
      End If

      LOCATION = './PUFFS_tseqs/srcKS_tseqs/'   ! Folder for puff tseq files
      TseqFile = 'SourceKS_'                    ! Puff source ensemble file
      LOCATION(18:19) = numb6(ks)(5:6)
      TseqFile(7:8)  = numb6(ks)(5:6)            ! File name of puff realizations
      Write (BlockID, '(I4.4,A1,I3.3)' ) 20*ib, '_', 20*jb
      FilePath = trim(LOCATION)//trim(TseqFile)//BlockID//'.lz'
     recl = (20*20*Ntms*Nrlz)*4    ! 1 block of raw time sequences ...
      Call lz4_read( 11, trim(FilePath), tseqs, recl, ISTAT )

      Return
      End Subroutine READ_PUFF_TSEQS

c  *******************************************************************.*********

      Subroutine READ_PLUME_XSECS ( xsecs, istep, NX, NY, Nsrc )

c  ___________________________________________________________________.__________

c  'READ_PUFF_xsecs' reads in the PLUME density cross-sections for all sources
c  for the single realization 'kr' (kr = 1 to Nrlz).  The results are put into
c  the real cross-section array 'xsecs.'  Density values in cells blocked out
c  by the building geometry have been set to hard zero in this dataset.  The
c  main function of this routine is to construct the correct file name and path
c  to retreive and decompress the set of Nsrc cross-sections identified by the
c  arguments.

c  xsecs    Real*4 array dimensioned NX x NY x Nsrc, i.e. 6 ground-level planes
c  istep    Timestep selected.  istep must be multiple of 25 for realization 1
c           (kr = 1) and a multiple 100 for the remaining 15 realizations.
c  NX       X extent of grid,  used for dimensioning xsecs  (here 1200)
c  NY       Y extent of grid,  used for dimensioning xsecs  (here 800)
c  Nsrc     Number of sources, used for dimensioning xsecs  (here 6)

c  Program written initially  Jay P. Boris  21 Jul 2013
c  Most recent modifications  Jay P. Boris  12 Dec 2016
c  ___________________________________________________________________.__________
```

```
c  Declare the main program data and associated local variables. . .
c  ___________________________________________________________________.__________
      Implicit  NONE
      Integer,intent(in)::  istep, NX, NY, Nsrc
      Real,intent(out)::    xsecs(NX,NY,Nsrc)    ! Density values at 'istep'

      Logical,save::        FirstCall = .true.
      Integer,save::        i, recl, JSTAT, Count = 0
      Character*80          LOCATION, XsecFile, FilePath
      Character*6,save::    numb6(0:999999)
c  ___________________________________________________________________.__________

c  Compute once and save a set of 6-digit character strings for naming files ...
c  ___________________________________________________________________.__________
      If ( FirstCall ) Then
         FirstCall = .false.
         Do i = 0, 999999
            Write ( numb6(i), '(I6.6)' ) i
         End Do
      End If

c  Now read the density cross-section for realization 'kr' for all Nsrc sources.
c  ___________________________________________________________________.__________
      If ( mod(istep,25) .ne. 0 ) Then
         Write (*,*) 'READ_PLUME_XSECS error: istep', istep
        Stop
      End if

      Count = Count + 1    ! How many times has the routine been called?
      LOCATION = './PLUMES_xsecs/'
      XsecFile = 'plumes_6srcs_'        ! Folder and cross-section file name
      FilePath = trim(LOCATION)//trim(XsecFile)//numb6(istep)//'.lz'
      recl = NX*NY*Nsrc*4               ! Nsrc density cross-sections in bytes
      If ( count.lt. 11 ) Write ( *,* )
     &   'READ_PLUME_XSECS: ', trim(FilePath), recl, istep
      Call lz4_read( 20, trim(FilePath), xsecs, recl, JSTAT )
      Close ( unit=20 )

      Return
      End Subroutine READ_PLUME_XSECS

c  *******************************************************************.*********

      Subroutine READ_PLUME_TSEQS ( tseqs, ks, ib, jb, Ntms )

c  ___________________________________________________________________.__________

c  'READ_PUFF_TSEQS' reads in a single 20-cell by 20-cell contiguous block of
c  puff density time sequences for source 'ks.'  All Nrlz realizations have been
c  collected into a single file.  The results are put real array 'tseqs.'
c  Density values in cells blocked out by the building geometry have been set to
c  hard zero in this dataset.  The main function of this routine is to construct
c  the correct file name and path to retreive and decompress the block of Nrlz
c  time-sequences identified in the argument list.

c  tseqs    Real*4 array dimensioned 20 x 20 x Ntms x Nrlz contains density time
c           sequences for 400 contiguous ground-level CFD cells located in block
c           ib, jb of the 2400 blocks covering the 1200 x 800 domain spanning
c           6 km by 4 km.  NOTE: In the plumes the time sequences is first index
```

```
c  ks       Source to be read, here ks = 1, ..., Nsrc = 6.  Since sources are in
c           different locations, different blocks(ib,jb) will have to be chosen
c           to obtain the same position relative to the different sources.  Thus
c           combining realizations makes sense for time sequences but combining
c           sources does not.
c  ib       Horizontal block index:  ib = (icell+19)/20   (ib = 1, ..., 60)
c  jb       Vertical block index:    jb = (jcell+19)/20   (jb = 1, ..., 40)
c           where icell and jcell are the indices of the CFD cell of interest.
c           Block ib,jb contains cells   icell = 1 + 20*(ib-1) through 20*ib
c           with jcell = 1 + 20*(jb-1) through 20*jb.
c  Ntms     Number of slices in entire time sequence (here 11401 (with slice 0 )
c  Nrlz     Number of puff realizations, used for dimensioning xsecs  (here 16)

c  Program written initially  Jay P. Boris  23 Jul 2013
c  Most recent modifications  Jay P. Boris  12 Dec 2016
c  ____________________________________________________________________._________
c  Declare the main program data and associated local variables. . .
c  ____________________________________________________________________._________
      Implicit  NONE

      Integer,intent(in)::  ks, ib, jb, Ntms
      Real,intent(out)::   tseqs(-10:Ntms,20,20)   ! src 'ks' time sequences

      Logical,save::       FirstCall = .true.
      Integer              i, recl, ISTAT
      Character*80::       LOCATION, TseqFile, FilePath
      Character*8::        BlockID
      Character*6,save::   numb6(0:999999)
c  ____________________________________________________________________._________

c  Compute once and save set of 6-digit character strings for naming files ...
c  ____________________________________________________________________._________
      If ( FirstCall ) Then
         FirstCall = .false.
         Do i = 0, 999999
            Write ( numb6(i), '(I6.6)' ) i
         End Do
      End If

c  Now read the time histories for src 'ks' for all Nrlz sources.
c  ____________________________________________________________________._________
      If ( ks.lt.1 .or. ks.gt.6 ) Then
         Write ( *,* ) 'READ_PLUME_TSEQS error: ks out of range ', ks
         Stop
      End If

      LOCATION = './PLUMES_tseqs/srcKS_tseqs/'   ! Folder for plume tseq files
      TseqFile = 'SourceKS_'                     ! Plume individual file name
      LOCATION(19:20) = numb6(ks)(5:6)
      TseqFile(7:8) = numb6(ks)(5:6)
      Write (BlockID, '(I4.4,A1,I3.3)' ) 20*ib, '_', 20*jb
      FilePath = trim(LOCATION)//trim(TseqFile)//BlockID//'.lz'
     recl = Ntms*20*20*4     ! 1 block of raw time sequences ...
      Call lz4_read( 11, trim(FilePath), tseqs, recl, ISTAT )

      Return
      End Subroutine READ_PLUME_TSEQS

c  ********************************************************************.*********
```

Appendix B: The READPLUMES Test and Verification Program

The READPLUMES test and verification program is appended immediately below. It may be possible to implement this program by extracting it directly out of this report but the file will also be included in the package containing the four datasets, geometry definition file, mask, utility ‘lz4’ subroutines, and verification output files.

```
C-*-fortran-*-
c  ********************************************************************.*********

      Program READPLUMES

c  ____________________________________________________________________.__________

c  READPLUMES is a program example illustrating the use of the 'lz4' utilities
c  provided for selecting, reading, and decompressing the continuous source
c  ground-level tracer density cross sections in dataset 'PLUMES_xsecs' and the
c  the run-long density time-sequences in dataset 'PUFFS_tseqs.'  Any one of the
c  Ntms time snapshots, which contains cross sections for all Nsrc sources, can
c  be retrieved using routine 'READ_PLUME_XSECS' as shown.  11401 cross sections
c  are recorded at intervals of 2.5 seconds, out to 28,850 seconds for each of
c  the Nsrc = 6 sources.  Each
c  realization of the release for all 6 sources is stored in a separate folder.
c  The first realization, with data contained in subfolder 'PUFFS_rlz01,' was
c  recorded every 2.5 seconds to allow the possibility of higher-frequency
c  analyses and thus contains 2401 cross sections rather than 601.

c  The entire 'PLUMES_xsecs' dataset has also been transposed to speed access
c  to density time histories at selected grid points without having to read all
c  52GB of data.  These full time sequences, 601 density values per grid point
c  at 10-second intervals, are collected into files that contain all Nrlz = 16
c  realizations together for a single source.  Each of the Nsrc = 6 sources has
c  a separate subfolder in this dataset, 'PUFFS_tseqs,' and is accessed using
c  'READ_PUFF_TSEQS.'  Because there are 960,000 grid points which would require
c  960,000 separate files, the time sequences have been combined into files
c  having 400 sequences for a 20x20 square block of contiguous CFD cells.  Each
c  block spans a 100m x 100m area.  There are 2400 such blocks across the grid
c  and thus there are 2400 time-sequence block files in each of 'PUFFS_tseqs'
c  6 source subfolders.

c  Below the two utility routines 'READ_PUFF_XSECS' and 'READ_PUFF_TSEQS' are
c  used to read a small amount of data from these two PUFFS datasets.  Short
c  diagnostic printouts are provided so a new user on another computer has a
c  easy way to check that the desired data are being accessed correctly.  These
c  tests are described briefly in the database documentation, NRL Memorandum
c  Report "Airborne Plume and Puff Databases for an Urban Landscape."

c  Program initially written  Jay P. Boris   3 Sep 2013
c  Most recent modifications  Jay P. Boris  12 Dec 2016
c  ____________________________________________________________________.__________

c  Declare the main program data and associated local variables. . .
c  ____________________________________________________________________.__________
      Implicit  NONE

      Character*80 MaskFile
      Character*6  num6(0:999999)
      Integer   NX, NY, Nsrc, Ntms  ! Dimensions of arrays
      Integer   ks, i, j, it, ISTEP, stepdel, IOS, ii, jj
```

```
      Integer   icu, ibu, jcu, jbu, icl, ibl, jcl, jbl
      Integer   isrc(6), jsrc(6)
      Real      delta_x, delta_t, time, rhou(11401,6), rhol(11401,6)
      Real      dt_phys, Xsize, Ysize, thresh, rhomax, BldgZero
      Real      Bmin,Bmax,  Gmin, Gmax   ! Range of height values
      Integer   NG0, NBG0, NBGNZ         ! Stats to check geometry
      Integer   NB0, NCT(6)

c  ___________________________________________________________________._________
      Integer*2,allocatable:: mask(:,:)
      Real, allocatable::  Bhgt(:,:), Ghgt(:,:)
      Real,allocatable::   xsecs(:,:,:), tseqsu(:,:,:), tseqsl(:,:,:)

c  Source locations:  src1  src2  src3  src4  src5  src6
      Data      isrc / 122,  240,  282,  338,  510,  844 /
      Data      jsrc / 326,  518,  290,  738,  398,  596 /

c  *******************************************************************.*********
c  Initialize the run with data defaults ...
c  *******************************************************************.*********
      Write ( *,* ) '  READPLUMES test program beginning execution: '
      Write ( *,* ) '   ' ! Blank line on output
      NX = 1200           ! Urban grid X (east) dimension at 5-meter resolution
      NY =  800           ! Urban grid Y (north) dimension at 5-meter resolution

      delta_t = 0.1       ! sec, CFD timestep, files recorded every 100 steps
      delta_x = 5.0       ! 5-meter resolution for basic 'urban' CFD runs
      Nsrc = 6            ! There are six sources in the data base ...
      Ntms = 11401        ! Number of time slices for plume density data
      stepdel = 25        ! Good for all realizations including the 1st.
      dt_phys = delta_t*stepdel  ! e.g. 10 second interval

      Xsize = delta_x*NX  ! Entire domain X size regardless of resolution
      Ysize = delta_x*NY  ! Entire domain Y size regardless of resolution
      thresh = 1.0E-8     ! Typical threshold value, depends on source strength

c  Compute a set of 6-digit character strings for naming files ...
c  ___________________________________________________________________._________
      Do i = 0, 999999
         Write ( num6(i), '(I6.6)' ) i
      End Do

c  Print out input data
c  ___________________________________________________________________._________
 1401 Format( '1. READPLUMES input: NX   NY   Nsrc Ntms '
     &        'stepdel   delta_x delta_t dt_phys ',
     &     /, 19X, 3I5, I6, 2X, I4, 3X, 3F8.3 )
      Write ( *, 1401 ) NX, NY, Nsrc, Ntms, stepdel,
     &                  delta_x, delta_t, dt_phys

c  Read a 1200 x 800 'mask' file of 2-byte positive integers that can be used
c  as a graphics mask to overlay the CFD grid geometry on cross-section plots.
c  ___________________________________________________________________._________
      Allocate ( mask(NX,NY), Bhgt(NX,NY), Ghgt(NX,NY) )  ! Geometry data
      Allocate ( xsecs(NX,NY,Nsrc) )                      ! Nsrc cross sections
      MaskFile  = 'urban_geometry_MASK'
      Open ( unit=12, file=trim(MaskFile), iostat=IOS, err=7003,
     &        status='unknown', access='direct', recl=2*NX )
      Go To 7013
```

```
 7003 Write ( 6, * ) 'OPEN error:  Mask file problems', IOS
      Pause " Mask Read Problem ???"

 7013 Do j = 1, NY
         Read ( 12, rec = j ) ( mask(i,j), i = 1, NX )
      End Do
      Close ( unit=12, status='keep' )
 1402 Format( '2. READPLUMES mask file:', A, '  size NX x NY ', 2I5 )
      Write ( *,* ) '   '  ! Blank line on output
      Write ( 6, 1402 ) trim(MaskFile), NX, NY

c  Read in the detailed geometry at the resolution actually used by FAST3D-CT.
c  ___________________________________________________________________._________
      Open ( unit=15, file='urban_geometry', iostat=IOS,
     &     status='unknown', form='unformatted' )
      Read( 15 ) Bhgt, Ghgt  ! Buildings and ground ...
      Close ( unit=15, status='keep' )

c  Compute building and ground (max and min) statistics as a verification check.
c  ___________________________________________________________________._________
      NB0 = 0;      NG0 = 0;    NBG0 = 0;     NBGNZ = 0
      Bmax = maxval( Bhgt );     Bmin = minval( Bhgt )
      Gmax = maxval( Ghgt );     Gmin = minval( Ghgt )
      Do j = 1, NY;   Do i = 1, NX
         If ( Bhgt(i,j) .eq. 0.0 ) NB0 = NB0 + 1
         If ( Ghgt(i,j) .eq. 0.0 ) NG0 = NG0 + 1
         If ( Bhgt(i,j).eq.0.0.and.Ghgt(i,j).eq.0.0 ) NBG0 = NBG0 + 1
         If ( Bhgt(i,j).gt.0.0.and.Ghgt(i,j).gt.0.0 ) NBGNZ = NBGNZ + 1
      End Do;         End Do

 1101 Format ( 1X, /,'3. Urban_geometry test results ...' )
 1102 Format ( '   Building max & min', 2(F7.2,'m'),
     &         '   Ground max & min', 2(F7.2,'m') )
 1103 Format ( '   #Bldg 0: ', I6, '  #Grnd 0: ', I6,
     &       '  #Both 0: ', I6, '  #Both NZ: ', I6 )
      Write ( *, 1101)
      Write ( *, 1102 ) Bmax, Bmin, Gmax, Gmin
      Write ( *, 1103 ) NB0, NG0, NBG0, NBGNZ

c  Plot the Bhgt and Ghgt arrays with mask using your favorite plot methods and
c  use the max and min values calculated above for scaling. e.g.
c  ___________________________________________________________________._________
c      Call REAL_XY_PLOT ( Bhgt, mask, NX, NY, Bmax, Bmin )
c      Call REAL_XY_PLOT ( Ghgt, mask, NX, NY, Gmax, Gmin )

c  *******************************************************************.*********
c  Test READ_PLUME_XSECS by calculating a table of areas contaminated above the
c  threshold 'thresh' for each of the source plumes as a function of time.  Do
c  the integral every 10 minutes for two hours for each plume.  Count the cells
c  and multiply by 25.0E-6 to get the area in square kilometers.
c  *******************************************************************.*********
      Write ( *, 3101 );    Write ( *, 3103 )   ! Label the columns
 3101 Format ( 1X, /,'4. Plume ground-level hazard areas (sq km)' )
 3102 Format ( F10.2, 1X, 1P6E11.3 )
 3103 Format ( '     time     Plume 1    Plume 2    Plume 3 ',
     &         '   Plume 4    Plume 5    Plume 6' )

      Do istep = 15000, 87000, 6000  ! Every 10 minutes for 2 hours
         time = delta_t * real(istep - 15000)
```

```
         Call READ_PLUME_XSECS( xsecs, istep, NX, NY, Nsrc )   ! rlz kr !
         NCT = 0  ! One counter for each source
         Do j = 1, NY;   Do i = 1, NX;   Do ks = 1, Nsrc
            If ( xsecs(i,j,ks) .ge. thresh ) NCT(ks) = NCT(ks) + 1
         End Do;         End Do;         End Do
         Write (*, 3102) time, ( 25.0E-6*real(NCT(ks)), ks = 1, Nsrc )
      End Do

c  EXAMPLE: Setting densities inside buildings to a fixed negative number
c  to distinguish the hard zero values in buildings from zero densities found
c  over open ground.  This may be useful for some statistics and diagnostics.
c  ___________________________________________________________________.__________
      bldgZero = -1.0   ! Example of a nonphysical building flag density ...
      Do j = 1, NY;   Do i = 1, NX
        If ( Bhgt(i,j) .gt. 0.0 ) xsecs(i,j,:) = BldgZero   ! Inside buildings
      End Do;         End Do

c  *******************************************************************.*********
c  Define each source location as the cell having maximum density two hours
c  after the sources begin.  These locations are fixed throughout the run and
c  these plumes will always be strongest in the vicinity of the release point.
c  For these datasets the plume source locations differ by two cells from the
c  puff releases.  An accident of history.
c  *******************************************************************.*********
      istep = 87000   ! Two hours ...
      Call READ_PLUME_XSECS( xsecs, istep, NX, NY, Nsrc )   ! rlz kr !
      Write ( *, 1410 ) istep
      Do ks = 1, Nsrc   ! Loop over the sources ...
         Write ( *, 1411 ) ks, isrc(ks), jsrc(ks)
         rhomax = -1.0
         Do j = 1, NY;        Do i = 1, NX
            If ( rhomax .lt. xsecs(i,j,ks) ) Then
               rhomax = xsecs(i,j,ks)
               isrc(ks) = i;  jsrc(ks) = j   ! save source locations
            End If
         End Do;              End Do
         Write ( *, 1412 ) ks, isrc(ks), jsrc(ks), istep, rhomax
      End Do  ! Loop over the Nsrc sources ...
 1410 Format ( 1X, /, '5. Source cell indices from step ', I6 )
 1411 Format ( '   Source ', I1, ' is at cell', 2I5,'  in FAST3D-CT' )
 1412 Format ( '   Source ', I1, ' is found @', 2I5,'  at step ', I6,
     &           '  rhomax= ', 1PE11.4 )

c  *******************************************************************.*********
c  At two 'user-chosen' grid points, read density time sequences for 2 hours.
c  These are the upper and lower points of the 6-plume figures.  This requires
c  reading 20x20 blocks of time sequences that include all Nsrc sources.  Block
c  indices are limited to ib = 1, ..., 60 and jb = 1, ..., 40.
c     Prepare a csv file of these time-dependent densities for Sources 1 to 5.
c  Source 6 is too far north for its plume to reach these points.
c  *******************************************************************.*********
      Allocate ( tseqsu(Ntms,20,20), tseqsl(Ntms,20,20) )   ! 1 source block
      icu = 850;   jcu = 250                        ! Upper (U) diagnostic cell
      ibu = (icu + 19)/20;   jbu = (jcu + 19)/20    ! Actual block # (U)
      icl = 850;   jcl = 150                        ! Lower (L) diagnostic cell
      ibl = (icl + 19)/20;   jbl = (jcl + 19)/20    ! Actual block # (U)
      Write ( *, * ) '    '   ! One blank line on output
      Write ( *, 1201)  ibu, jbu, ibl, jbl
 1201 Format ( '6. Testing READ_PLUME_TSEQS for sources 1 to 5:',
```

```
     &   ' output a .csv file',
     &   /, '   upper cell (850, 250) found in block', 2I3,
     &   /, '   lower cell (850, 250) found in block', 2I3 )


c  Open the .csv file for data to prepare density time trace plots.
c  ___________________________________________________________________.__________
      Open ( unit=7, file=trim('Upper&Lower_rhot.csv'), iostat=IOS,
     &       err=4003, status='unknown', form='formatted' )
      Write ( *,* ) '  Opened Upper&Lower_rhot.csv on unit 7'
      Go To 4005
 4003 Write ( *,* ) 'Error in opening Upper&Lower_rhot.csv', IOS
 4005 Write ( *,* ) '     '  ! blank line on output
      Write ( 7, 4008 ) ! Put labels on the .csv file ...
      Write ( *, 4009 )  ! Label the printed output ...
      Do ks = 1, Nsrc
         Call READ_PLUME_TSEQS ( tseqsu, ks, ibu, jbu, Ntms )
         Call READ_PLUME_TSEQS ( tseqsl, ks, ibl, jbl, Ntms )
         Do istep = 15000, 87000, 50      ! Every 5 seconds out to 2 hours
            it = 1 + (istep - 15000)/25   ! Dataset slice number for istep
            time = (istep - 15000) * delta_t
            ii = icu - 20*(ibu-1);   jj = jcu - 20*(jbu-1);   ! Block indices
            rhou(it,ks) = tseqsu(it,ii,jj)
            ii = icl - 20*(ibl-1);   jj = jcl - 20*(jbl-1);   ! Block indices
            rhol(it,ks) = tseqsl(it,ii,jj)
         End Do
      End Do  ! Loop over the 6 sources ...

c  Now print the results ...
      Do istep = 15000, 87000, 50      ! Every 5 seconds out to 2 hours
         it = 1 + (istep - 15000)/25   ! Dataset slice number for istep
         time = (istep - 15000) * delta_t
         Write (7, 4006) time, ( max(1.0e-12, rhou(it,i)), i = 1, 5 ),
     &                         ( max(1.0e-12, rhol(it,i)), i = 1, 5 )
         If ( mod(istep-15000,6000) .eq. 0 )   ! Every 10 minutes ...
     &      Write ( *, 4007 ) time, ( rhou(it,i), i=1,5 ),
     &                              ( rhol(it,i), i=1,5 )
      End Do
 4006 Format ( F9.2,',', 10(1PE11.3,',') )
 4007 Format ( F10.2,  '  upper pt', 5(1PE11.3), /,
     &         10X,    '  lower pt', 5(1PE11.3) )
 4008 Format ('time (seconds),Plume 1 Upper,Plume 2 Upper,',
     &         'Plume 3 Upper,Plume 4 Upper,Plume 5 Upper,',
     &         'Plume 1 Lower,Plume 2 Lower,Plume 3 Lower,',
     &         'Plume 4 Lower,Plume 5 Lower,' )
 4009 Format ('      time             Plume 1    Plume 2 ',
     &        '   Plume 3    Plume 4    Plume 5' )
       Close ( unit=7, status='keep' )

c  *******************************************************************.*********
      Write ( 6, * ) '  READPLUMES: End of Computations '

      Stop
      End  Program READPLUMES

c  *******************************************************************.*********
```

Appendix C: The READPUFFS Test and Verification Program

The READPUFFS test and verification program is appended immediately below. It may be possible to implement this program by extracting it directly out of this report but the file will also be included in the package containing the four datasets, geometry definition file, mask, utility 'lz4' subroutines, and verification output files.

```
C-*-fortran-*-
c  *******************************************************************.*********

      Program READPUFFS

c  ___________________________________________________________________.__________

c  READPUFFS is a program example illustrating the use of the 'lz4' utilities
c  provided for selecting, reading, and decompressing the acute (instantaneous
c  source) ground-level density cross sections in dataset 'PUFFS_xsecs' and the
c  the run-long density time-sequences in dataset 'PUFFS_tseqs.'  Any one of the
c  Ntms time slices, which contains density cross sections for all sources, can
c  be retrieved using routine 'READ_PUFF_XSECS' as shown.  There are 601 time
c  slices, recorded at intervals of 10 seconds, out to 100 minutes after each
c  tracer release for all Nrlz = 16 separate realizations of each source.  Each
c  realization of the release, including all sources, is stored in a separate
c  folder.  The first realization, contained in subfolder 'PUFFS_rlz01,' was
c  recorded every 2.5 seconds to allow the possibility of higher-frequency
c  analyses.  Thus 'PUFFS_rlz01' contains 2401 cross sections rather than 601.

c  The entire 'PUFFS_xsecs' dataset has also been transposed to speed access to
c  density time histories at selected grid points without having to read all
c  52 GB of data.  These time sequences, 601 density values per grid point
c  at 10-second intervals, are collected into files that contain all Nrlz = 16
c  realizations together for a single source.  Each of the Nsrc = 6 sources has
c  a separate subfolder in dataset 'PUFFS_tseqs,' and is accessed using routine
c  'READ_PUFF_TSEQS.'  Because 960,000 grid points would require 960,000 files,
c  each with 16x6 time sequences, the sequences have been combined into files
c  having 400 sequences each in a 20x20 square block of contiguous CFD cells.
c  Each block spans a 100m x 100m area.  There are 2400 such blocks across the
c  grid.  Thus there are 2400 time-sequence block files in each of 'PUFFS_tseqs'
c  6 source subfolders.

c  Below the two utility routines 'READ_PUFF_XSECS' and 'READ_PUFF_TSEQS' are
c  used to read a small amount of data from these two PUFFS datasets.  Short
c  diagnostic printouts are provided so a new user on another computer has a
c  easy way to check that the desired data are being accessed correctly.  These
c  tests are described briefly in the database documentation, NRL Memorandum
c  Report "Airborne Plume and Puff Databases for an Urban Landscape."

c  Program initially written   Jay P. Boris    8 Nov 2012
c  Most recent modifications   Jay P. Boris   12 Dec 2016
c  ___________________________________________________________________.__________

c  Declare the main program data and associated local variables. . .
c  ___________________________________________________________________.__________
      Implicit  NONE

      Character*80 MaskFile
      Character*6  num6(0:999999)
      Logical   inside                    ! .true. when i,j inside a building
      Integer   NX, NY, Nsrc, Nrlz, Ntms  ! Dimensions of arrays
```

```
      Integer   ks, kr, i, j, it, ISTEP, stepdel, IOS, ib, jb, ii, jj
      Integer   icell, jcell, msrc, itime, isrc(6), jsrc(6)
      Real      delta_x, delta_t, time
      Real      rhomax, rhosum, rhosqr, rhorms, downwind
      Real      dt_phys, Xsize, Ysize, thresh, xsv
      Real      rhoavg, rhoavp, rhoavm, BldgZero
      Real      Bmin,Bmax,  Gmin, Gmax   ! Range of height values
      Integer   NG0, NBG0, NBGNZ         ! Stats to check geometry
      Integer   NB0, NBNZ, NO0, NTH, NGT

c  ___________________________________________________________________._________
      Integer*2,allocatable:: mask(:,:)
      Real, allocatable::     Bhgt(:,:), Ghgt(:,:)
      Real,allocatable::      xsecs(:,:,:), tseqs(:,:,:,:)
      Real,allocatable::      rho(:,:)

c  Source locations:  src1  src2  src3  src4  src5  src6
      Data      isrc / 124,  242,  284,  340,  512,  846 /
      Data      jsrc / 326,  518,  290,  738,  398,  596 /

c  *******************************************************************.*********
c  Initialize the run with data defaults ...
c  *******************************************************************.*********
      Write ( *,* ) '  READPUFFS test program beginning execution: '
      Write ( *,* ) '   ' ! Blank line on output
      NX = 1200           ! Urban grid X (east) dimension at 5-meter resolution
      NY =  800           ! Urban grid Y (north) dimension at 5-meter resolution

      delta_t = 0.1       ! sec, CFD timestep, files recorded every 100 steps
      delta_x = 5.0       ! 5-meter resolution for basic 'urban' CFD runs
      Nsrc = 6            ! There are six sources in the data base ...
      Ntms = 601          ! Number of times for density data sets, all rlz
      Nrlz = 16           ! There are currently 16 puff realizations
      stepdel = 100       ! Good for all realizations including the 1st.
      dt_phys = delta_t*stepdel  ! e.g. 10 second interval

      Xsize = delta_x*NX  ! Entire domain X size regardless of resolution
      Ysize = delta_x*NY  ! Entire domain Y size regardless of resolution
      thresh = 1.0E-8     ! Typical threshold value, depends on source strength

c  Compute a set of 6-digit character strings for naming files ...
c  ___________________________________________________________________._________
      Do i = 0, 999999
         Write ( num6(i), '(I6.6)' ) i
      End Do

c  Print out input data
c  ___________________________________________________________________._________
 1401 Format( '1. READPUFFS input: NX   NY   Nsrc Ntms Nrlz '
     &    'stepdel  delta_x delta_t dt_phys ',
     &     /, 18X, 5I5, 2X, I4, 3X, 3F8.3 )
      Write ( *, 1401 ) NX, NY, Nsrc, Ntms, Nrlz, stepdel,
     &                  delta_x, delta_t, dt_phys

c  Read a 1200 x 800 'mask' file of 2-byte positive integers that can be used
c  as a graphics mask to overlay the CFD grid geometry on cross-section plots.
c  ___________________________________________________________________._________
      Allocate ( mask(NX,NY), Bhgt(NX,NY), Ghgt(NX,NY) )  ! Geometry data
      Allocate ( xsecs(NX,NY,Nsrc) )                      ! Nsrc cross sections
```

```
      MaskFile  = 'urban_geometry_MASK'
      Open ( unit=12, file=trim(MaskFile), iostat=IOS, err=7003,
     &        status='unknown', access='direct', recl=2*NX )
      Go To 7013
 7003 Write ( 6, * ) 'OPEN error:  Mask file problems', IOS
      Pause " Mask Read Problem ???"

 7013 Do j = 1, NY
         Read ( 12, rec = j ) ( mask(i,j), i = 1, NX )
      End Do
      Close ( unit=12, status='keep' )
 1402 Format(1X, /, '2. READPUFFS mask file:', A, '   NX x NY ', 2I5 )
      Write ( 6, 1402 ) trim(MaskFile), NX, NY

c  Read in the detailed geometry at the resolution actually used by FAST3D-CT.
c  ___________________________________________________________________._________
      Open ( unit=15, file='urban_geometry', iostat=IOS,
     &     status='unknown', form='unformatted' )
      Read( 15 ) Bhgt, Ghgt  ! Buildings and ground ...
      Close ( unit=15, status='keep' )

c  Compute building and ground (max and min) statistics as a check ...
c  ___________________________________________________________________._________
      NB0 = 0;      NG0 = 0;    NBG0 = 0;     NBGNZ = 0
      Bmax = maxval( Bhgt );    Bmin = minval( Bhgt )
      Gmax = maxval( Ghgt );    Gmin = minval( Ghgt )
      Do j = 1, NY;   Do i = 1, NX
         If ( Bhgt(i,j) .eq. 0.0 ) NB0 = NB0 + 1
         If ( Ghgt(i,j) .eq. 0.0 ) NG0 = NG0 + 1
         If ( Bhgt(i,j).eq.0.0.and.Ghgt(i,j).eq.0.0 ) NBG0 = NBG0 + 1
         If ( Bhgt(i,j).gt.0.0.and.Ghgt(i,j).gt.0.0 ) NBGNZ = NBGNZ + 1
      End Do;         End Do

 1101 Format ( 1X, /,'3. Urban_geometry test results ...' )
 1102 Format ( '   Building max & min', 2(F7.2,'m'),
     &         '   Ground max & min', 2(F7.2,'m') )
 1103 Format ( '   #Bldg 0: ', I6, '  #Grnd 0: ', I6,
     &       '  #Both 0: ', I6, '  #Both NZ: ', I6 )
      Write ( *, 1101)
      Write ( *, 1102 ) Bmax, Bmin, Gmax, Gmin
      Write ( *, 1103 ) NB0, NG0, NBG0, NBGNZ

c  Plot the Bhgt and Ghgt arrays with mask using your favorite plot methods and
c  the max and min values calculated above for scaling. e.g.
c  ___________________________________________________________________._________
c      Call REAL_XY_PLOT ( Bhgt, mask, NX, NY, Bmax, Bmin )
c      Call REAL_XY_PLOT ( Ghgt, mask, NX, NY, Gmax, Gmin )

c  *******************************************************************.*********
c  Test READ_PUFF_XSECS by reading user-chosen 'PUFFS_rlz' file containing 6
c  density cross-sections into 'xsecs', one for each source, at the specified
c  timestep 'istep' and realization 'kr.'  This example acts as a model for new
c  users in accessing the 'PUFFS_xsec' database.  Using the data at step 15100,
c  10 sec after release, this example shows how to estimate source locations.
c  *******************************************************************.*********
      kr = 16          ! Test the kr-th realization ...
      istep = 15100   ! 10.0 sec after initialization
      Call READ_PUFF_XSECS( xsecs, istep, kr, NX, NY, Nsrc )
```

```
c  EXAMPLE: Setting densities inside buildings to a fixed negative number
c  to distinguish the hard zero values in buildings from zero densities found
c  over open ground.  This may be useful for some statistics and diagnostics.
c  ___________________________________________________________________._________
      bldgZero = -1.0   ! Example of a nonphysical building flag density ...
      Do j = 1, NY;   Do i = 1, NX
        If ( Bhgt(i,j) .gt. 0.0 ) xsecs(i,j,:) = BldgZero  ! Inside buildings
      End Do;         End Do

c  Compute some counts on the puff density data just read for each source ...
      Write ( *, 1113 )
      Do ks = 1, Nsrc
         NB0 = 0;        NBNZ = 0
         NO0 = 0;        NTH = 0;       NGT = 0
         Do j = 1, NY;   Do i = 1, NX
            inside = Bhgt(i,j).gt.0.0;  xsv = xsecs(i,j,ks)
            If ( inside .and. xsv.eq.BldgZero ) NB0 = NB0 + 1
            If ( inside .and. xsv.ne.BldgZero ) NBNZ = NBNZ + 1
            If ( .not.inside .and. xsv.eq.0.0 ) NO0 = NO0 + 1
            If ( .not.inside .and. xsv.gt.0.0
     &                       .and. xsv.le.thresh ) NTH = NTH + 1
            If ( .not.inside .and. xsv.gt.thresh ) NGT = NGT + 1
         End Do;         End Do  ! Loops over the full CFD grid
         Write ( *, 1112 ) ks, NB0, NBNZ, NO0, NTH, NGT
 1112    Format( '   Source ', I1, ' #Bldg0 & NZ',
     &      I7, I5,  '   #Open0, <TH & >TH', I7, 2I6 )
 1113    Format( 1X, /, '4. Bldg Zero, nonzero,',
     &      ' and Open Zero, < threshold, and >= threshold',
     &      ' counts:' )
      End Do

c  *******************************************************************.*********
c  Define each source location as the cell having maximum density shortly after
c  the puff's release.  This location is fixed throughout the run though puffs
c  will blow away from the source quickly in open areas.  For this database
c  Source locations:  src  i    j   src  i    j   src  i    j
c   (cell indices)     1  124  326   2  242  518   3  284  290
c                      4  340  738   5  512  398   6  846  596
c  *******************************************************************.*********
      Write ( *, 1410 ) istep
      Do ks = 1, Nsrc  ! Loop over the sources ...
         Write ( *, 1411 ) ks, isrc(ks), jsrc(ks)
         rhomax = -1.0
         Do j = 1, NY;        Do i = 1, NX
            If ( rhomax .lt. xsecs(i,j,ks) ) Then
               rhomax = xsecs(i,j,ks)
               isrc(ks) = i;  jsrc(ks) = j   ! save source locations
            End If
         End Do;              End Do
         Write ( *, 1412 ) ks, isrc(ks), jsrc(ks), istep, rhomax
      End Do  ! Loop over the Nsrc sources ...
 1410 Format ( 1X, /, '5. Source cell indices from step ', I6 )
 1411 Format ( '   Source ', I1, ' is at cell', 2I5,'  in FAST3D-CT' )
 1412 Format ( '   Source ', I1, ' is found @', 2I5,'  at step ', I6,
     &          '  rhomax= ', 1PE11.4 )

c  *******************************************************************.*********
c  Read a user-chosen 20x20 block of time sequences for source 'ks' that has
c  all Nrlz=16 realizations.  ib = 1, ..., 60 and jb = 1, ..., 40 are allowed
```

```
c  block indices.
c  *******************************************************************.*********
      Allocate ( tseqs(Ntms,Nrlz,20,20) )
      Allocate ( rho(Ntms,Nrlz) )

c  Prepare a csv file of density values at a point down wind from source msrc.
c  Also print a small table of 'spot check' densities for each source.
c  ___________________________________________________________________._________
      downwind = 500.0 ! meters  from the source; should see some thrashing
      msrc = 5  ! Chosen because 500 meters downwind is relatively open!
      Do ks = 1, Nsrc     ! Each source has a different downwind station
         icell = isrc(ks) + 0.866*(downwind/5.0)    ! Wind from 300-deg
         jcell = jsrc(ks) - 0.500*(downwind/5.0)
         ib = (icell+19)/20;   jb = (jcell+19)/20   ! Actual 20x20 block number
         Write ( *, 1201)  ks, icell, jcell, ib, jb
 1201    Format ( 1X,/, '6. Testing READ_PUFF_TSEQS for source', I2, /,
     &      '   icell,jcell indices', 2I5,
     &      '  are found in 20x20 block ib,jb ', 2I5 )

         Call READ_PUFF_TSEQS ( tseqs, ks, ib, jb, Ntms, Nrlz )

c  EXAMPLE: Setting densities inside buildings to a fixed negative number
c  to distinguish the hard zero values in buildings from zero densities found
c  over open ground.  This may be important for some statistics and diagnostics.
c  ___________________________________________________________________._________
         bldgZero = -1.0   ! Example of a nonphysical building flag density ...
         Do jj = 1, 20;   Do ii = 1, 20
            j = jj + 20*(jb - 1);    i = ii + 20*(ib - 1)         ! Cell indices
            If ( Bhgt(i,j).gt.0.0 ) tseqs(:,:,ii,jj) = BldgZero   ! Inside bldgs
         End Do;          End Do

c  Compute some counts of the time sequences just read for source ks ...
         Write ( *, 1114 )
 1114    Format( '   Bldg Zero, nonzero, and Open Zero, ',
     &          '< threshold, and >= threshold counts:' )
         NB0 = 0;        NBNZ = 0
         NO0 = 0;        NTH = 0;       NGT = 0
         Do jj = 1, 20;    Do ii = 1, 20
            j = jj + 20*(jb - 1);    i = ii + 20*(ib - 1)        ! Cell indices
         Do it = 1, Ntms, 60;  Do kr = 1, Nrlz     ! Only check tabulated times
            inside = Bhgt(i,j).gt.0.0;  xsv = tseqs(it,kr,ii,jj)
            If ( inside .and. xsv.eq.BldgZero ) NB0 = NB0 + 1
            If ( inside .and. xsv.ne.BldgZero ) NBNZ = NBNZ + 1
            If ( .not.inside .and. xsv.eq.0.0 ) NO0 = NO0 + 1
            If ( .not.inside .and. xsv.gt.0.0
     &                       .and. xsv.le.thresh ) NTH = NTH + 1
            If ( .not.inside .and. xsv.gt.thresh ) NGT = NGT + 1
         End Do;          End Do  ! Loop over it and kr
         End Do;          End Do  ! Loops over i and j
         Write ( *, 1112 ) ks, NB0, NBNZ, NO0, NTH, NGT

c  Copy all Ntms by Nrlz density values into a temporary array ...
         ii = icell - 20*(ib-1)  ! Location of icell in block ib,jb
         jj = jcell - 20*(jb-1)  ! Location of jcell in block ib,jb
         rho(:,:) = tseqs(:,:,ii,jj)

c  Print a few diagnostic numbers for each source from the first 6 realizations
c  every 600 steps (= 1 min intervals) up to 600 seconds ...
 1030    Format ( 1X, /, '   Spot check density values for Source ',
```

```
     &       I1, '  in cell ', 2I5, '  Bhgt ', F7.2 )
         Write ( *, 1030 ) ks, icell, jcell, Bhgt(icell,jcell)
         Write ( *, * ) '  time(s)    rlz 1      rlz 2      rlz 3',
     &      '      rlz 4      rlz 5      rlz 6 '
         Do itime = 1, 61, 6    ! 0, 1, 2, ..., 10 min
            time = dt_phys*real(itime-1)  ! Elapsed time in seconds ...
            Write ( *, 1031 ) time, ( rho(itime,kr), kr = 1, 6 )
        End Do
 1031   Format ( F9.2, 1X, 1P6E11.3 )

c  Compute the average and standard deviation of the Nrlz ensemble density and
c  write out a .csv (comma separated values) file for subsequent plotting when
c  considering source 'msrc.'
c  ____________________________________________________________________._________
         If ( msrc .eq. ks ) Then
            Open ( unit=7, file=trim('Source5_rhot.csv'), iostat=IOS,
     &         err=4003, status='unknown', form='formatted' )
            Write ( *,* ) 'Opened Source5_rhot.csv on unit 7'
            Go To 4005
 4003       Write ( *,* ) 'Error in opening Source5_rhot.csv', IOS
 4005       Continue
 4008       Format ('time (seconds),rlz 1,rlz 2,rlz 3,rlz 4,rlz 5,',
     &              'rlz 6,rlz 7,rlz 8,rlz 9,rlz 10,rlz 11,rlz 12,',
     &              'rlz 13,rlz 14,rlz 15,rlz 16,',
     &              'avg/rms+,average,avg + rms,' )
            Write ( 7, 4008 )
         End If

         Do itime = 1, Ntms               ! Print entire time seq for all rlz
            time = dt_phys*real(itime-1)  ! Elapsed time in seconds ...

            rhosum = 0.0
            Do kr = 1, Nrlz
               rhosum = rhosum + rho(itime,kr)
            End Do
            rhoavg = max(1.0E-12, rhosum/real(Nrlz) )  ! At least 1e-12
            rhosqr = 0.0
            Do kr = 1, Nrlz
               rhosqr = rhosqr + (rho(itime,kr) - rhoavg)**2
            End Do
            rhorms = sqrt( rhosqr/real(Nrlz) )     ! This can be zero
            rhoavp = max( 1.0E-12, rhoavg + rhorms )
            rhoavm = rhoavg*(rhoavg/rhoavp)   ! Same ratio lower ...
            rhoavm = max(1.0E-12, rhoavm)     ! Keep lower bound .ge. 1.0E-12

            If ( msrc .eq. ks )
     &         Write ( 7, 1021 ) time, ( max(1.0E-12,rho(itime,kr)),
     &            kr = 1, Nrlz ), rhoavm, rhoavg, rhoavp

         End Do  ! Loop over Ntms time slices ...
 1021    Format ( F9.2,',', 19(1PE11.3,',') )
         If ( msrc .eq. ks ) Close ( unit=7, status='keep' )
      End Do  ! Loop over the Nsrc sources

c  ********************************************************************.*********
      Write ( 6, * ) '  READPUFFS: End of Computations '
      Stop
      End  Program READPUFFS
c  ********************************************************************.*********
```