A Better Way to Illustrate Atmospheric Dispersion in the Classroom

Phil Dacunto and Mike Hendricks
Department of Geography and Environmental Engineering
United States Military Academy, West Point, NY

Abstract. Students enrolled in introductory air pollution courses typically have difficulty fully understanding the impacts of changing variables in the Gaussian plume equation. It is challenging for students to visualize how the *entire plume* changes as the result of modifications in atmospheric stability, for example, since they typically calculate only one or two values in x, y, and z. Spreadsheets can help with completing these calculations quickly, but unfortunately can only plot in one dimension, and the results are still difficult to visualize. To address these limitations a Geographic Information System (GIS) based custom application was developed that coupled ESRI's ArcMap 9.1 with Matlab. Using inputs of stack height, wind speed, atmospheric stability, and source emission rate, the application creates an array of downwind ground level plume concentrations that are plotted onto a city map. The sum of these concentrations on the city's features such as schools are calculated. Though this application creates only a simplified model of the atmospheric dispersion process, it proves valuable in instruction since it is simple enough for students to understand, while at the same time demonstrating the dispersion process' "big picture." Students in an introductory air pollution course not only effectively employed the application to visualize the impact of changing plume variables, but also applied it to an optimization scenario that required repeated calculations of the downwind effects of various plumes to meet given concentration guidelines. After using this plume modeling application in the course, students had a better understanding of plume behavior, a better understanding of the value of information technology in solving engineering problems, and a greater interest in applications of atmospheric dispersion modeling.

INTRODUCTION

Most basic air pollution courses cover atmospheric dispersion and typically model concentrations of downwind pollutants with a Gaussian plume. Students traditionally calculate concentrations by hand at individual x, y, and z locations within the plume. This technique, however, makes it difficult for students to visualize the effects of changing variables such as stack height, wind speed, or source emission rate, on the geographic variation of pollutant concentrations. Spreadsheet programs such as Microsoft Excel can be employed to quickly calculate a large number of point concentrations. However, a spreadsheet's graphing capability is limited, and students still cannot effectively visualize the geographic variation of plume concentrations.

At the United States Military Academy, we address this shortfall in our introductory air pollution engineering course by leveraging the Department of Geography and Environmental Engineering's unique combination of disciplines. The Department includes not only an Environmental Engineering group, but also a Geospatial Information Science (GIS) group. Combining expertise from both groups, we built a custom application merging Matlab and ArcGIS that enabled students to visualize the cause and effect relationships between changing input variables and downwind plume concentrations. Using inputs of stack height, wind speed, atmospheric stability, and source emission rate, the application creates a georeferenced grid of plume concentrations aligned to a reference map. This approach provides the

additional benefit of increasing student exposure to information technology (IT), which helped the Environmental Engineering group to achieve the Academy's IT goals.

The custom application employs a simplified model of actual dispersion processes and does not account for all variables. This, however, is also its strength – students can understand the simple model and see the impacts of the most important variables on atmospheric dispersion. As a result, the application proves valuable in instruction since it quickly demonstrates the "big picture." In addition, students are able to use the application not only in visualizing the impact of changing plume variables, but also in an optimization scenario that requires repeated calculations of the downwind effects of plumes to meet given concentration guidelines.

After using the plume modeling application in the course, students had a better understanding of plume behavior, a better understanding of the use of information technology, and a greater interest in applications of atmospheric dispersion modeling. Such applications can help greatly in illustrating atmospheric dispersion in the classroom, and we would recommend them to anyone teaching a basic course on air pollution.

METHODOLOGY

The custom application consists of two components: one to calculation downwind concentrations, and another to plot those concentrations on a map and calculate impacts. Due to its flexibility in programming applications, and its ability to handle large matrices of data, we chose Matlab to perform the calculations of downwind plume concentrations. We chose ESRI's ArcMap 9.1 for visualization and analysis. We created the custom application so that students would be able to use it without previous experience with either program.

Calculating Downwind Concentrations - Matlab

The algorithm employed in the Matlab application calculated downwind, ground level concentrations using the Gaussian plume equation (Cooper and Alley, pg 612)[1]:

$$C = \frac{Q}{2\pi u \sigma_y \sigma_z} \exp\left(-\frac{y^2}{2\sigma_y^2}\right)\left\{\exp\left(\frac{(z-H)^2}{2\sigma_z^2}\right) + \exp\left(-\frac{(z+H)^2}{2\sigma_z^2}\right)\right\}$$

Q = source emission rate [g/s]
u= wind speed [m/s]
y = crosswind distance from stack of point of interest [m]
z = vertical height of point of interest (0 for ground-level concentration)[m]
H =effective stack height [m] (includes plume rise)
σ_y =horizontal stability parameter (a function of downwind distance x, and stability) [m]
σ_z =vertical stability parameter (a function of downwind distance x, and stability) [m]

This equation assumes total reflection of pollutants from the ground, but does not account for an inversion above the effective stack height that would cause additional reflection. By convention, the base of the source stack was at x = y = z = 0:

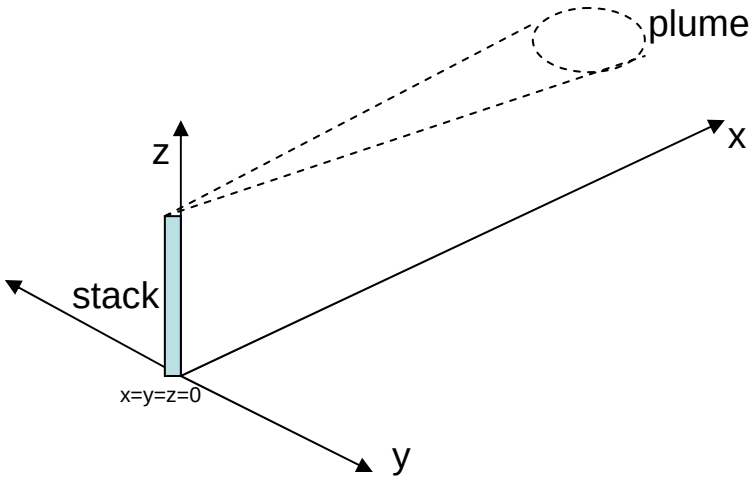

Figure 1: Coordinates for Gaussian plume equation

The Matlab algorithm takes inputs of wind speed [m/s], pollutant source strength [g/s], stack height [m], and stability class (A-F). Students entered these values, along with the name of their output file, in a specified portion of the program (figure 2).

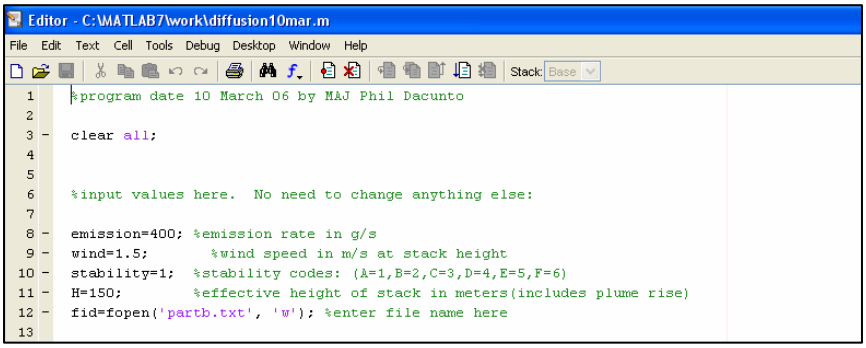

Figure 2: Matlab interface for setting variables

The algorithm employs Pasquill-Gifford curve fits from Turner (1969)[2] to estimate values of the horizontal and vertical stability parameters:

$$\sigma_y = e^{(I_y + J_y \ln(x) + K_y (\ln(x))^2}$$

$$\sigma_z = e^{(I_z + J_z \ln(x) + K_z (\ln(x))^2}$$

The I, J, and K coefficients vary by stability condition A-F (Seinfeld and Pandis, Table 18.2, original from Turner, 1969).[3]

The critical portions of the algorithm are two nested loops that use the program inputs listed above to calculate a matrix of downwind concentrations. For every value of y (the "outer loop"), the program calculates a number of values of x (the "inner loop"). Our application uses a fixed cell size of 10 meters. We felt that 10 meters provides a good balance between the necessary details to visualize the results while still minimizing excessive computations. Our application calculates values in an extent from -1000 meter to +1000 meters in the y direction, and from 0 to 10,000 meter in the x direction. Given the

10 meter cell size, the algorithm generates 200,000 concentration values. Cell size, as well as the limits of the system, can be changed by the user, but we preset the values for the students.

The output file from Matlab is formatted to enable easy importing into ArcMap. The file's header includes the number of columns and rows of the extent (in our case typically 1000 columns by 200 rows), the x and y location of the lower left corner, cell size, and a value for cells not calculated. The concentration values follow this header information (figure 3).

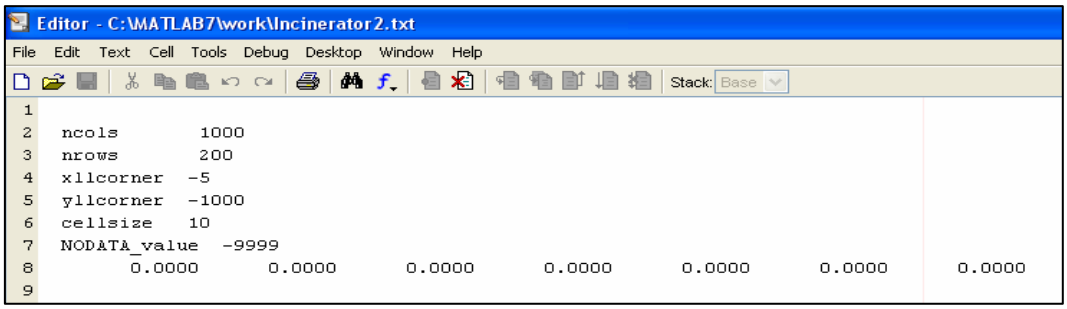

Figure 3: Matlab output file format

Notice that the initial concentrations are all zeroes, because these first values are on the edge of the plume, where concentrations are negligible. Scrolling to the right through the 200,000 values, however, one would see cells with concentrations greater than zero.

Visualizing and Analyzing Results - ArcGIS

A text file of concentration values is, of course, almost useless by itself, since it does not enable students to see the spatial relationship of concentration values. A Geographic Information System (GIS) is the perfect tool to visualize and analyze the distribution of plume concentration values. We chose to employ ESRI's ArcMap GIS application, a component of the company's ArcGIS suite of software tools. ArcMap is a powerful GIS that we could customize to allow students without GIS experience to import plume concentrations from MatLab, visualize those results, and calculate concentration values on various parts of the cities, such as schools. ArcMap includes a visual programming environment called Model Builder. With model builder we quickly generated tools and user interfaces to: (1) import Matlab concentration files and summarize concentration values over key areas of the city, and (2) combine plumes into an overall plume.

After generating plume concentrations in Matlab, students import these results into ArcMap. The tool's interface required students to provide the location by manually typing the proposed pollution source's UTM map coordinates, along with the name of both the input and output file (figure 4).

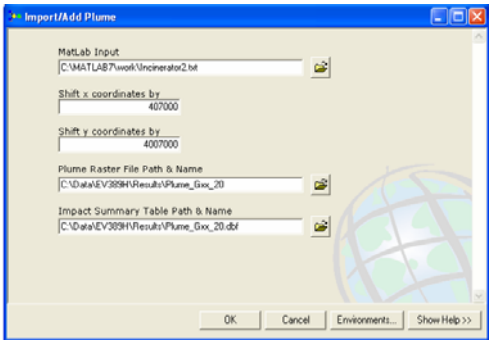

Figure 4: Plume import dialog box

The tool produces an overlay of pollutant concentration values on a map of the area of interest, in our case, a map of Arbil, Iraq (figure 5).

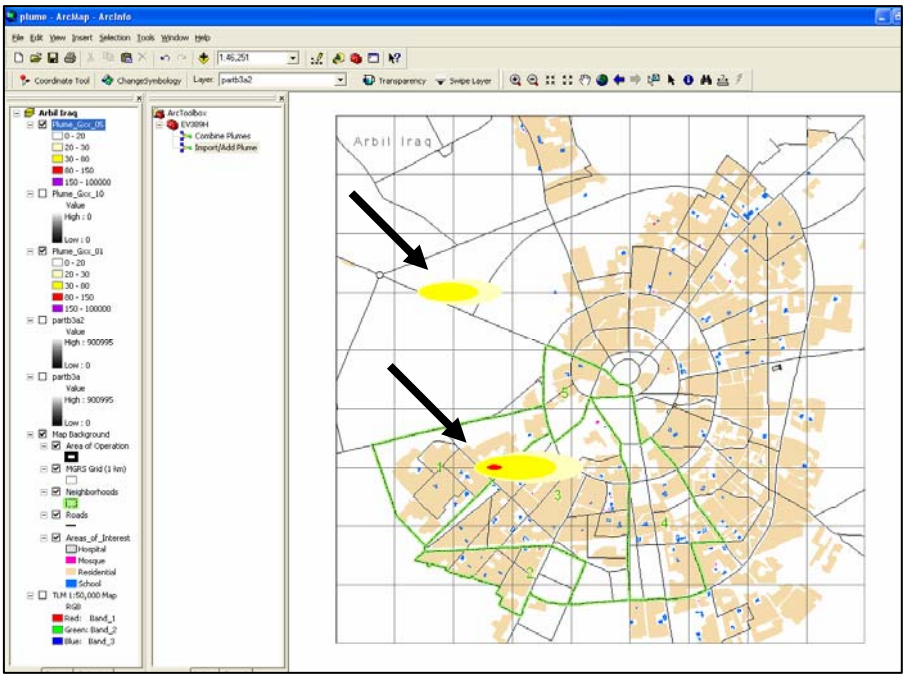

Figure 5: Pollutant contours plotted on the map

After importing and plotting a plume, students open a summary table that indicates concentration value statistics over different portions of the map, such as residential areas vs. industrial areas (figure 6). This table allows students to assess the impact of the plume and compare how these impacts change with different point source locations and characteristics, such as stack height.

Attributes of PartB1a

OID	BLDG_TYPE	ZONE_CODE	COUNT	AREA	MIN	MAX	RANGE	MEAN	STD	SUM
0	Residential	1	55166	5516600	11.6548	201.347	189.692	57.51	33.7668	3172600
1	School	2	259	25900	22.1809	231.065	208.884	50.3611	34.4622	13043.5
2	Mosque	3	26	2600	23.1132	95.8863	72.7731	50.3445	22.1729	1308.96

Figure 6: Pollutant impact summary table

A second tool and user interface allows students to combine plumes. This tool uses “map algebra,” which allows raster data sets to be combined on a cell by cell basis with an algebraic expression (figure 7). This tool enables students to visualize the effects of not just one plume, but the combined effects of several as they work through an analysis of a local air quality plan. A region that may have been below an air quality standard as the result of the effects of one plume may become an area of concern when plumes overlap. This is a critical tool that students needed to use as they worked on their dispersion projects.

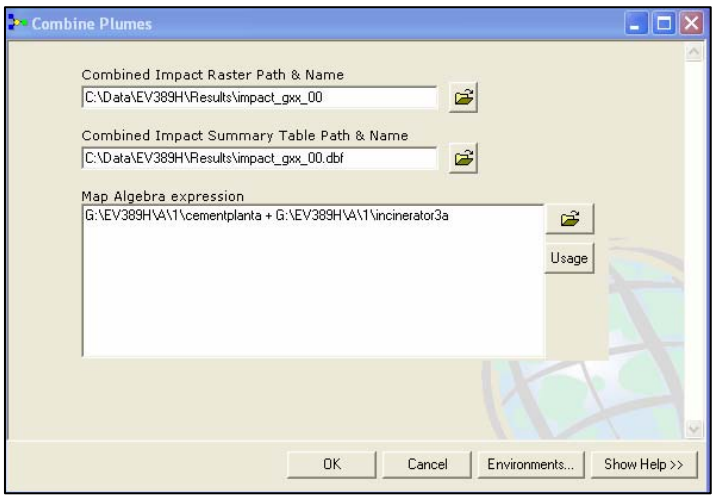

Figure 7: Plume combination dialog box

STUDENT PROJECTS

After being introduced to Matlab and ArcMap, students began work on an atmospheric dispersion project. There were two objectives for the project: (1) enable students to visualize the cause and effect relationships inherent in the Gaussian plume model (what happens when we increase stability, for example); and (2) familiarize students with the use of information technology, particularly networks, to solve a problem. The project consisted of both an individual and group component. The individual component focused on objective one, while the group portion was oriented on both objectives. Students conducted the project during work periods in our department's Geospatial Sciences Laboratory, which contains twenty networked workstations loaded with Matlab and ArcMap.

In the individual portion of the project, students first used the Gaussian plume equation to calculate concentrations by hand at specified locations. Subsequently, they used the computer applications to plot the entire plume, and also to verify their hand calculated results. This enabled them to see the "bigger picture" of the plume as a whole, instead of just one data point. In addition, students changed the effective stack heights, wind speeds, and stability conditions in order to visualize the plume's response to these varying input parameters.

We designed the group portion to be much more calculation-intensive so that we could not only help students visualize and understand plumes, but also demonstrate the power of networked IT resources in solving problems. In this portion, students worked in teams on an optimization scenario that involved the creation of a local air quality plan in the city of Arbil, Iraq, which was fictionally rebuilding its infrastructure through new factories and other industry that would have an impact on local air quality. We focused on the pollutant PM-10 (particulate matter 10 μm or less in diameter), and gave each group an unchanging set of stability (summer day with a few broken clouds) and wind conditions (from the west at 2.5 m/s).

We divided the city of Arbil into five fictional neighborhoods (numbered below in figure 8). Teams had four or five members, each responsible for the air quality of one neighborhood. One student functioned also as the group leader and "mayor" of Arbil, and had the responsibility of combining all of the local air quality plans into a plan that benefited the city as a whole.

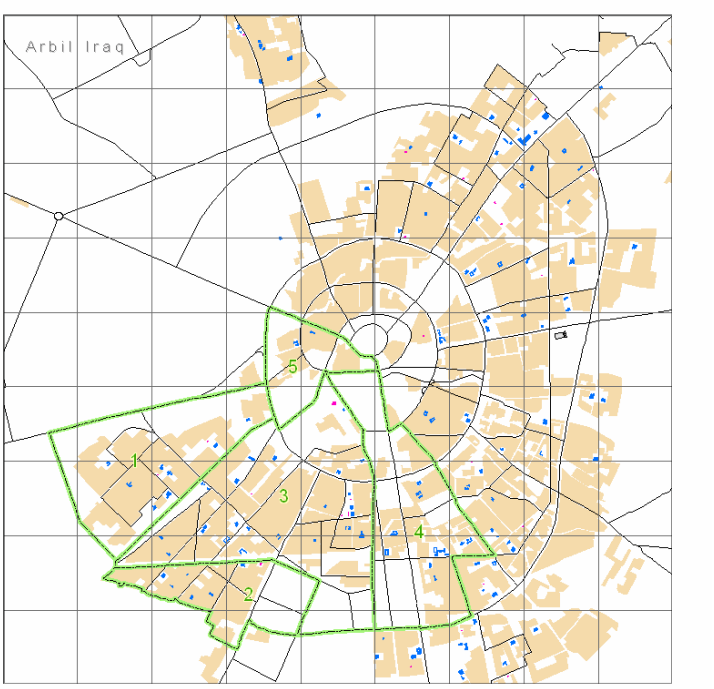

Figure 8: Fictional neighborhoods in city

To meet given air quality standards in their neighborhood, students determined the location, stack height, and emissions rate for three air pollution sources. Increases in stack heights, or reduction of baseline emissions through the installation of pollution control devices cost money, however, so students were not free to build a stack so high that pollutants would never reach the ground in Arbil (or reduce pollutants so extensively that plumes became nonexistent, despite how nice this would be in reality). The optimum solution, then would be the one that gave the best air quality (i.e., met all of the air quality standards) for the least cost. Students came up with a rudimentary plan based upon terrain analysis and wind direction, but refinement of that plan required extensive trial and error in the neighborhood as they moved individual sources and then combined plumes to achieve the best overall results for their neighborhood.

The best plan, however, for one neighborhood was not necessarily the best for other neighborhoods, especially those downwind. Thus, once each student determined the optimum plan for their neighborhood, the team worked together under the mayor's direction to further refine the plan so the combined plan was optimized for the city as a whole. This involved another round of analysis and trial and error, as well as the combined computing power of a network of 4-5 computers to achieve success. It was in this portion of the project that we hoped to achieve our second objective of familiarizing students with the use of networked IT systems to solve a problem.

RESULTS/DISCUSSION

Students picked up the mechanics of the application quickly, and came up with some good solutions (figure 9). Surprisingly, while there was no one "right" answer, the total optimized cost of successful plans was rather close. Unsuccessful projects were generally satisfactory at the neighborhood level, but did not successfully integrate neighborhood plans into a successful one for the city as a whole.

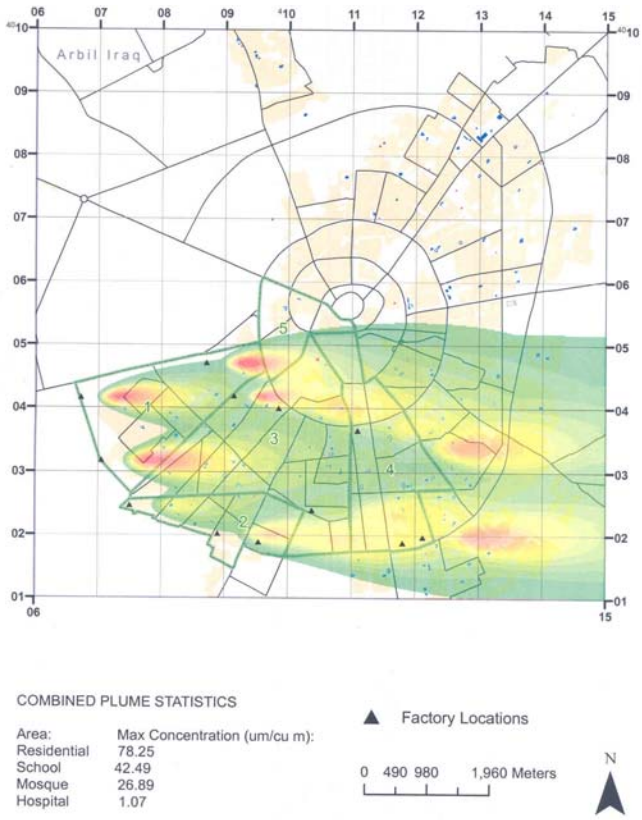

Figure 9: City overall plume

.

While our model is very simple in terms of predictions of pollutant concentrations (it only considers one pollutant from only a few sources under one set of meteorological conditions, for example), it worked extremely well as a tool for teaching about the Gaussian plume and information technology networks.

Prior to the creation of these applications, students' experience with the Gaussian plume was limited to the calculation of only a few values. While they discussed in class the effects of changes in each variable of the equation, students were not able to *visualize* these impacts on the plume as a whole, a valuable help to any type of learner. In addition, in some ways the project resembled a computer game, as teams worked together to "beat" it. This assisted with student motivation.

A downside of this "gaming" aspect to the applications was that students expected it to perform like a typical computer game. Thus, some commented that the applications were too "clunky;" i.e., it took too many different steps in two different programs to insert a new plume or modify the existing characteristics of a current one. Applications also created a large number of input and output files, each of which had to be managed on the server by each group in a specific way to avoid confusion between files. In some ways, this was a strength of the application, however; by avoiding the creation of a "black box" system that magically turned inputs into a plume, students were continually reminded of the assumptions and theory on which the model was based as they worked through each separate step in the process of creating the plume.

The project was equally valuable in demonstrating the value of IT networks to solving problems. Statistics from a student survey administered at the end of the course supported this, with 85% of responses marking "agree" or "strongly agree" for the following statement:

> *"The block of lessons we spent discussing the use of computers in air quality prediction, as well as the dispersion project, helped me to understand how computers can be used together to solve an air pollution problem."*

File management, the sharing of calculations by different computers, and the sheer number of computations involved all helped students to understand the value of the use of IT in solving engineering problems.

CONCLUSION

In the future, we intend to modify the ArcMap application so that it does the computational work of the Matlab application as well. If done correctly, this will streamline the processes somewhat without preventing the students from understanding what the program is doing and how it works. In addition, we would like to change the ArcMap application so that plume locations can be designated by mouse-click, vs. hand entry of source coordinates. Even in its current form, however, the set of applications was very valuable to the students' understanding of atmospheric dispersion and thus the course.

Based on our success with these applications, we would recommend them for any introductory air pollution engineering course that studies atmospheric dispersion. Students should start with computations by hand, and then graduate to these computer applications that can model the plume as a whole. The pedagogical strength of these applications lies in their simplicity – students can clearly see the cause and effect relationships between changing inputs and the plume. In addition to its value as a tool to teach about atmospheric dispersion, the program helps to integrate IT networks and resources into an engineering curriculum, and requires students to work together as a team. Thus, it is a valuable addition to any air pollution course or engineering curriculum.

Bibliography

1. Cooper, David and F.C. Alley (2002) *Air Pollution Control*. Waveland Press, Prospect Heights, IL.

2. Turner, D.B. (1969) *Workbook of Atmospheric Diffusion Estimates*, USEPA 999-AP-26. U.S. Environmental Protection Agency, Washington, D.C.

3. Seinfeld, John and Spyros Pandis (1998) *Atmospheric Chemistry and Physics*. John Wiley and Sons, New York, NY.