6.4 COUPLED RECEPTOR/DISPERSION MODELING WITH A GENETIC ALGORITHM

Sue Ellen Haupt *
The Pennsylvania State University, State College, PA

1. INTRODUCTION

Comparing transport and dispersion model output to monitored pollutant concentrations is difficult. Transport models often forecast ensemble averages while receptors record temporal averages. Depending on the averaging time, these two averages can differ widely when the transporting flow is turbulent (Wyngaard 1992, Wyngaard and Peltier 1996, Venkatram and Wyngaard 1998). Likewise, receptor models can be used to estimate the apportionment to potential sources if a sufficient number of chemical species has been measured. These models exhibit similar sources of uncertainty.

Receptor models are formulated to begin with pollutant information monitored at a receptor and to look backward, using data on several species and information about relative concentrations of those species from potential sources, to apportion the pollutant to the potential sources. In contrast, the chemical transport, or dispersion, models start with the source characteristics and use physics, mathematical, and chemical calculations to predict pollutant concentration at some distance from the source. Important input for those models includes information about the emissions from the source, the local atmospheric conditions, and some geographical characterization. Both types of models have been highly developed and forms of them are widely used for prediction and diagnosis of events.

It would be of great benefit to be able to couple the strengths of the two types of models. The transport and dispersion models use physics and chemistry principles to predict the statistical likelihood of concentrations of pollutants for given conditions. Receptor models are grounded in reality, taking actual measurements and estimating the fraction contribution from each source, without any consideration of the prevailing physics. A coupled model would combine the physical basis of the calculations with actual monitored pollutant concentrations.

There is a need to integrate the current technology of the forward-looking transport and dispersion models with backward-looking receptor models to define the apportionment of monitored data to their sources and to estimate the uncertainty involved. Here we couple a simple dispersion model with a chemical mass balance (CMB) receptor model in a way that optimizes source apportionment factors to determine the sources of monitored pollutant. We demonstrate the technique here using synthetic data.

A few investigators have used information on dispersion or chemical transport in computing source apportionment. Qin and Oduyemi (2003) began apportioning sources of PM_{10} with a receptor model, then augmented it with dispersion model predictions from vehicle emission sources. Cartwright and Harris (1993) used a genetic algorithm (GA) to apportion sources to pollutant data at receptors. The work of Loughlin, et al (2000) coupled an air quality model with receptor principles using a GA to design better control strategies to meet attainment of the ozone standard while minimizing total cost of controls at over 1000 sources. Recently, Haupt (2004, also reported in Haupt and Haupt 2004) has used a basic Gaussian plume model used in a similar manner to that described below, coupled with a chemical mass balance receptor model via a genetic algorithm to compute the source

Corresponding author address: Sue Ellen Haupt, Applied Research Laboratory, P.O. Box 30, Pennsylvania State University, State College, PA 16804;
e-mail: haupts2@asme.org

calibration factors necessary to best match the measured pollutant.

2. PROBLEM FORMULATION

The CMB receptor model is often used to apportion monitored concentrations received at receptors to the expected sources. It requires receptor data of different monitored species and known emission fractions for each of those species from a number of sources. The CMB model can be written as:

$$C \bullet S = R \tag{1}$$

where *C* is the source concentration profile matrix, which denotes the fractional emission of each species from a given source; R is the concentration of each species measured at a given receptor, and S is the unknown apportionment vector. A fit to the data produces the fraction contribution from each source, *S*. Our coupled approach replaces the emission fractions in *C* with concentrations predicted by a transport model. Thus *S* becomes a calibration factor for the transport model dispersed emissions for an ensemble of time periods relative to the actual concentrations at the receptor. The modeled concentrations are time dependent, on a timescale that matches the meteorological variations. The receptor data is also time dependent, but on a timescale that matches the monitoring sample length. The source calibration vector (*S*) must be optimized to account for the time varying weather and emission rates.

For *N* sources for *M* time periods, matrix equation (1) can be shown in expanded form:

$$\begin{bmatrix} C_{11} & C_{12} & \cdots & C_{1n} & \cdots & C_{1N} \\ C_{21} & & & & & \\ \vdots & & & & & \\ C_{m1} & & & \ddots & & \\ \vdots & & & & & \\ C_{m1} & & & & & C_{MN} \end{bmatrix} \begin{bmatrix} S_1 \\ S_2 \\ \vdots \\ S_n \\ \vdots \\ S_N \end{bmatrix} = \begin{bmatrix} R_1 \\ R_2 \\ \vdots \\ R_m \\ \vdots \\ R_M \end{bmatrix} \tag{2}$$

where: C_{mn} = the modeled contribution of source *n* at for time period *m*

S_n = the unknown calibration factor for source *n*

R_m = the monitored particulate concentration at the receptor for time period *m*

As long as $M \geq N$, *S* can be computed by standard techniques (matrix inversion if $M = N$ or optimization otherwise).

For our simple illustration, to compute the elements in the modeled concentration matrix C, we will use Gaussian plume dispersion:

$$C_{mn} = \frac{Q_{mn}}{u\sigma_z\sigma_y 2\pi} \exp\left(\frac{-y_{mn}^{\ 2}}{2\sigma_y^{\ 2}}\right) \left[\exp\left(\frac{-(z_r - H_e)^2}{2\sigma_z^{\ 2}}\right) + \exp\left(\frac{-(z_r + H_e)^2}{2\sigma_z^{\ 2}}\right) \right] \tag{3}$$

where: C_{mn} = concentration of emissions from source *n* over time period *m* at a receptor

(x, y, z_r) = Cartesian coordinates of the receptor in the downwind direction from the source.

Q_{mn} = emission rate from source *n* over time period *m*

u = wind speed

H_e = effective height of the plume centerline above ground

σ_y, σ_z = standard deviations of the concentration distribution in the y and z directions, respectively.

For our simple example, we'll compute the standard deviations following Beychock (1994).

$$\sigma = \exp\left[I + J(\ln(x) + K(\ln(x))^2 \right] \tag{4}$$

where x is the downwind distance (in km) and I, J, and K are empirical coefficients dependent on the Pasquill Stability Class, which depends on wind speed, direction, and insolation and can be looked up in tables (Beychok 1994). The concentrations

computed in this manner from each source form the C_{mn} in equation (2).

The source calibration factor serves to optimize agreement between the transport model and the receptor observations. That factor can be interpreted as an error or uncertainty in the modeling process in comparison to the monitored data. This uncertainty comes from the input data and from the modeling process itself. The primary sources of error could be characterized as: 1. the source emission rate; 2. the accuracy and representativeness of the meteorological input, both in terms of directly measured variables such as wind speed and direction, as well as in derived quantities such as mixing height (representing the boundary layer depth) and atmospheric stability characterization; 3. the model's characterization of the atmospheric dispersion and chemical transformations; 4. not correctly modeling the stochastic fluctuations due to turbulence; and 5. errors in the monitoring data.

3. SOLUTION METHOD

The remaining issue is how to best optimize the fit between the modeled dispersion and the monitored receptor data. This involves computing the best calibration factor, *S*. Because the optimization software must be robust and able to deal with ill conditioned problems and potentially complex solution spaces, we choose methods from the artificial intelligence community. The applications presented here use genetic algorithms (GAs) to perform the optimization. GAs are well suited to many optimization problems where more traditional methods fail. Some of the advantages they have over conventional numerical optimization algorithms are that they:

- Optimize with continuous or discrete parameters,
- Don't require derivative information,
- Simultaneously search from a wide sampling of the objective function surface,
- Deal with a large number of parameters,
- Are well suited for parallel computers,
- Optimize parameters with extremely complex objective function surfaces,
- Provide a list of semi-optimum parameters, not just a single solution,
- May encode the parameters so that the optimization is done with the encoded parameters, and
- Works with numerically generated data, experimental data, or analytical functions.

These advantages outweigh the GAs' lack of rigorous convergence proofs.

There are many breeds of GA that are discussed in detail in Haupt and Haupt (2004). Here we apply a continuous parameter GA that is, one in which the parameters are real numbers. The flow chart in Figure 1 provides a "big picture" overview of a continuous GA. The parameters are the genes which are strung together in a one-dimensional array known as a chromosome. The GA begins with a population of chromosomes which are fed to the cost function for evaluation. The fittest chromosomes survive while the highest cost ones die off. This process mimics natural selection in the natural world. The lowest cost survivors mate. The mating process combines information from the two parents to produce two offspring. Some of the population experiences mutations.

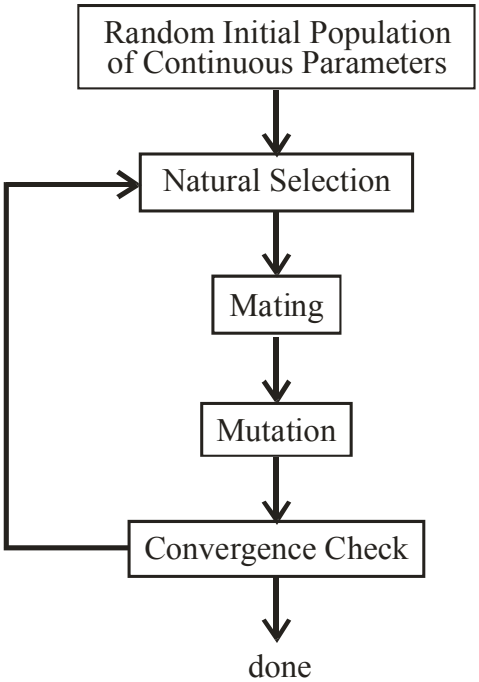

Figure 1. Flowchart of continuous parameter genetic algorithm.

As seen in the figure, the first step of a continuous parameter genetic algorithm is creating the population of chromosomes. First, the real parameters are concatenated together into a chromosome as:

$$chromosome = \left[p_1 p_2 \cdots p_\alpha \cdots p_{N_{par}} \right] \quad (5)$$

where the p_i are the parameters and there are a total of N_{par} parameters. The parameters are simply floating point numbers. The encoding function needs only keep track of which digits represent which parameters and to make sure they are within given bounds. A population of such chromosomes is created using a random number generator so that the chromosome arrays are gathered together in a two dimensional matrix. Once the chromosomes have been created, their cost or fitness must be evaluated. This is done by the cost or objective function, which is very problem specific. The lowest cost chromosomes (N_{keep}) remain in the population while the higher cost ones are deemed less fit and die off. The reduced population is then the portion of the population available for mating.

There are a variety of methods to pair the chromosomes for mating. Some popular methods are reviewed by Haupt and Haupt (2004). Here, we choose to pair the chromosomes according to numerical rank. After the cost function evaluation, the chromosomes are sorted in order from lowest cost to highest. That is, the nth chromosome will have a probability of mating of:

$$P_n = \frac{N_{keep} - n + 1}{\sum_{k=1}^{N_{keep}} k} \quad (6)$$

Then the cumulative probabilities are used for selecting which chromosomes mate.

Once two parents are chosen, some method must be devised to produce offspring which are some combination of these parents. Many different approaches have been tried for crossing over in continuous parameter genetic algorithms (Adwuya 1996, Haupt and Haupt 2004).

The method used here is a combination of an extrapolation method with a crossover method. We wanted to find a way to closely mimic the advantages of the binary genetic algorithm mating scheme. It begins by randomly selecting a parameter in the first pair of parents to be the crossover point.

$$\alpha = roundup\left\{ random \times N_{par} \right\} \quad (7)$$

Wel let

$$\begin{aligned} parent_1 &= \left[p_{m1} p_{m2} \cdots p_{m\alpha} \cdots p_{mN_{par}} \right] \\ parent_2 &= \left[p_{d1} p_{d2} \cdots p_{d\alpha} \cdots p_{dN_{par}} \right] \end{aligned} \quad (8)$$

where the *m* and *d* subscripts discriminate between the *mom* and the *dad* parent. Then the selected parameters are combined to form new parameters that will appear in the children:

$$\begin{aligned} p_{new1} &= p_{m\alpha} - \beta \left[p_{m\alpha} - p_{d\alpha} \right] \\ p_{new2} &= p_{d\alpha} - \beta \left[p_{m\alpha} - p_{d\alpha} \right] \end{aligned} \quad (9)$$

where β is also a random value between 0 and 1. The final step is to complete the crossover with the rest of the chromosome as before:

$$\begin{aligned} offspring_1 &= \left[p_{m1} p_{m2} \cdots p_{new1} \cdots p_{dN_{par}} \right] \\ offspring_2 &= \left[p_{d1} p_{d2} \cdots p_{new2} \cdots p_{mN_{par}} \right] \end{aligned} \quad (10)$$

If the first parameter of the chromosomes is selected, then only the parameters to right of the selected parameter are swapped. If the last parameter of the chromosomes is selected, then only the parameters to the left of the selected parameter are swapped. This method does not allow offspring parameters outside the bounds set by the parent unless β is greater than one. In this way, information from the two parent chromosomes is combined a way that mimics the crossover process during meiosis.

If care is not taken, the genetic algorithm converges too quickly into one region of the cost surface. If this area is in the region of the global minimum, that is good. However, some functions have many local minima and the algorithm could get stuck in a local well. If we do nothing to solve this tendency to converge

quickly, we could end up in a local rather than a global minimum. To avoid this problem of overly fast convergence, we force the routine to explore other areas of the cost surface by randomly introducing changes, or mutations, in some of the parameters. A mutated parameter is replaced by a new random parameter.

The cost function was formulated to minimize the difference between the two sides of (1), summed over the total number of meteorological periods considered. This normalized residual is:

$$\text{resid} = \frac{\sqrt{\sum_{m=1}^{M} (C \bullet S - R)^2}}{\sqrt{\sum_{m=1}^{M} (R)^2}} \qquad (11)$$

where M is the total number of meteorological periods. Note that each meteorological period might be composed of averages over many shorter periods. Thus, the GA evaluates the summation cost function for each random chromosome of parameters for each iteration. In spite of the large number of cost function evaluations, CPU time remains modest. For the poorly conditioned problems we often encounter in real data, we have found that the GA works better at minimizing the matrix equation than competing techniques. This cost is also the metric used below to compare the performance for different cases.

4. APPLICATION

4.1 Synthetic Data on a Circle

We validate the coupled receptor/dispersion model technique by testing it on carefully constructed synthetic data. The initial test cases place a receptor at the origin and 16 sources in a circle of radius 500 m spaced every 22.5 degrees (see Figure 2). Receptor data is created using the same dispersion model to be used for the coupled model optimization. To fit data for 16 sources, we need more than 16 independent meteorological periods. Meteorological data were created to represent wind directions from 16 points of the wind rose and representative wind speeds. These initial runs assume stability D for ease of comparison. The dispersion model was run using one hour averaging over the meteorological data and using assumed calibration factors, *S*, that we hope to match with the coupled model. Thus, we have used a dispersion model to create receptor data that matches the source/receptor configuration and the synthetic meteorological data.

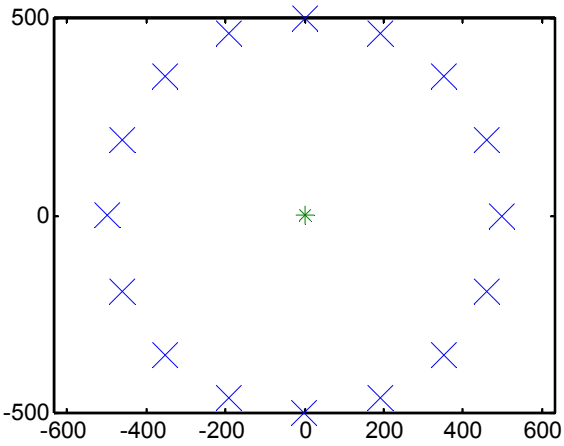

Figure 2. Configuration for circular synthetic data. Sources are denoted by "X" and the receptor as "*".

The coupled receptor/dispersion model was then run with the synthetically generated data. The first tests were done by setting the calibration factors to 0, except for one source that was set to a 1. The genetic algorithm, when run with a sufficient number of iterations, will gravitate toward the correct solution. For this problem, the number of iterations determine the smallness of the residual. Figure 3 demonstrates the GA convergence over 50,000 iterations. We see that the GA continues to minimize the residual. However, a MATLAB run with this many iterations takes approximately 2.9 hours of CPU time on a 2.4 GHz dual processor Xeon machine. The next question is how many iterations are necessary to get an acceptably small residual with a reasonable amount of computer time. Table 1 shows the results of a sensitivity study of residuals versus the number of iterations. Since the GA uses random numbers, a different residual is expected for each run that uses a different random seed. Thus, for this table, each configuration was run 5 times and the mean and standard deviation of the

residuals is listed. To assure that this analysis is not unique to the single source being turned on, it was repeated for a two source configuration, with source numbers 1 and 4 each having calibration factors of 0.5. Those results are reported in Table 2.

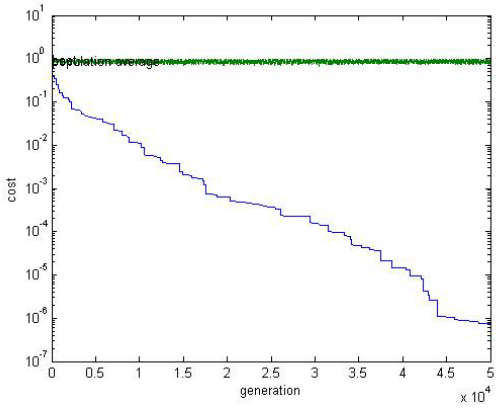

Figure 3. Convergence history of 50,000 iterations of a GA for the single source configuration. The green line gives mean residual and the blue line is the best residual.

Table 1. Residual size as a function of the number of GA iterations.

Iters	Best Residual	Mean Residual	Stand Dev	CPU time (s)
500	0.179	0.269	0.096	109
1000	0.155	0.191	0.030	232
2000	0.052	0.077	0.034	339
5000	0.034	0.052	0.020	1078
50000 *	5.36×10^{-4}			10313

- based on a single run

Table 2. Residual as a function of the number of GA iterations for a two source configuration.

Iters	Best Residual	Mean Residual	Stand Dev	CPU time (s)
1000	0.065	0.120	0.037	228
2000	0.047	0.088	0.028	432
5000	0.017	0.029	0.015	1154

In addition, one more sensitivity to the number of iterations was run for four sources each having a calibration value of 0.25. Figure 4 compares the actual solution to that computed by single runs of the GA for several differing iterations. We see that even for a relatively small number of iterations, the solutions compare relatively well. The more iterations, the better the comparison with the know calibration factors.

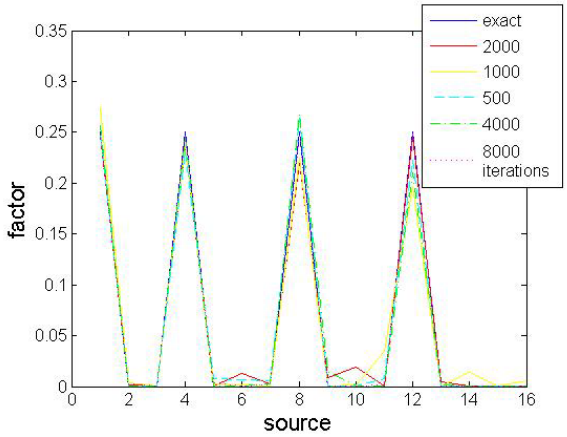

Figure 4. Demonstration of GA convergence for varying number of iterations for representative runs using four sources of 16.

4.2 Actual Emission Configuration with Synthetic Meteorological Data

A second comparison is done using an actual emission configuration. Data was obtained for Cache Valley, Utah. A map of the sources is given in Figure 4. The actual location of the sources was obtained from the state of Utah emission inventory and heights were estimated. However, at this stage, each source was still assigned the same artificial emission rate. The receptor location is the actual receptor on Main Street in Logan, Utah. For verification purposes, the meteorological data was produced synthetically to systematically sample the wind field. Emission factors were assumed, once again assigning a factor of 0 to all sources and turning one or more sources back on to validate the methodology.

Using real source configuration is a much more difficult problem than placing sources in

a concentric circle. For instance, consider the case where two sources might be at the same angle from the receptor but at different distances. If the wind speed was not variable, it would be difficult to distinguish between the contribution from those two sources. Thus, more variability in meteorological conditions is more likely to produce a correct allocation of source apportionment factors.

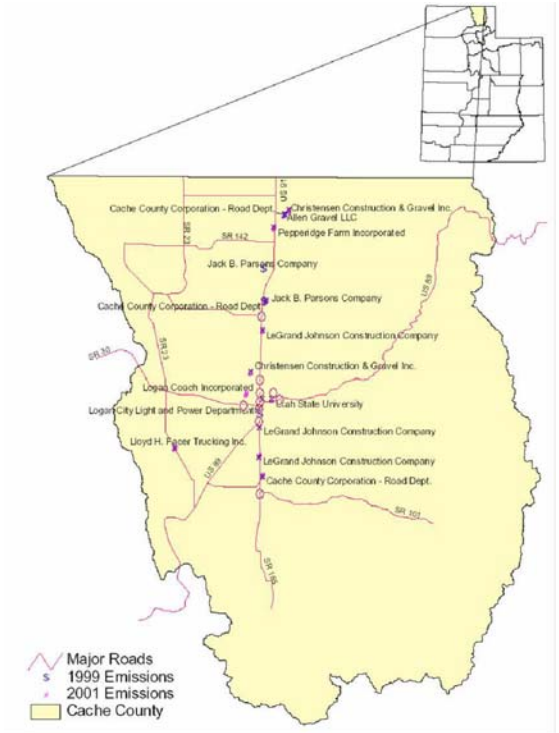

Figure 4. Map of emission sources in Cache Valley, UT. Receptor is denoted by "*".

The first validation example is with only one source factor of 1 used to create the data and none of the other sources contributing. The source chosen was 1.5 km west of the receptor, a direction unlike the remaining sources. As seen in Figure 5, the algorithm correctly identified the correct source in 10,000 iterations. Source 4 had some spurious contribution implied, but that source is 25 km away, so its contribution would be well dispersed by the time it reached the receptor. The residual for this run was 0.0047604.

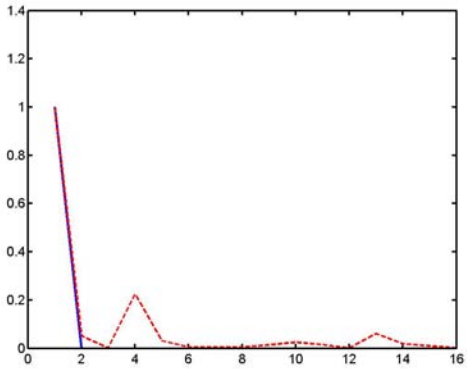

Figure 5. Results of a 10,000 iteration source optimization for Cache Valley, Utah. A single source was initialized. The solid blue line is the exact data created synthetically. The dashed red line is the optimized solution.

An second example run was done with three sources tuned to an apportionment factor of 1 while the rest are given zeros. The apportionment factors were optimized using the coupled model using 64 meteorological periods and 10,000 iterations. The comparison appears as Figure 6. The three sources that were given 1's are well captured. However, another five sources are spuriously assigned large apportionment factors, in spite of the relatively small residual of 0.070144. Sources 2, 4, 5 and 12, which were mistakenly assigned large apportionment factors, are 23 to 25 km away from the receptor. Thus, their contribution is likely to totally disperse by the time it reaches the receptor given the simple Gaussian plume model used. Therefore, those apportionment factors are meaningless anyhow. Source 12 is 8.5 km away and therefore likely to contribute, but it is in the same direction as the three sources that are making a real contribution. The lack of directional distinction makes it difficult to correctly identify only those sources that contribute to receptor pollutant concentration with the current configuration of the coupled model.

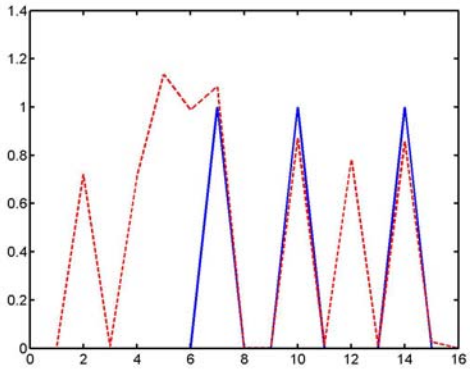

Figure 6. Results of a 10,000 iteration source optimization for Cache Valley, Utah. The solid blue line is the exact data created synthetically. The dashed red line is the optimized solution.

5. DISCUSSION

Although this work is preliminary, it demonstrates the utility of an optimization approach to couple forward looking dispersion models with backward looking receptor models. We have demonstrated that for circularly symmetric source/receptor configurations, the model can correctly identify a single source or some combination of sources that is expected to contribute to the total pollutant monitored at a receptor when the synthetic data is created using the same dispersion model and meteorological data as used in the coupled model. In addition, when an actual configuration is used with synthetic emissions and meteorological data, the coupled model can still identify a single source. However, there is more difficulty in correctly identifying multiple sources. When the direction of the source from the receptor is not independent from other sources or the source is sufficiently far away, it is easy to misidentify additional sources.

These results assume a very simple dispersion model. More work needs done on calibrating this source/receptor model technique before it is useful for actual source identification. Some of the first elements to be added are to work with variable source strengths and use actual meteorological data to estimate atmospheric stability and the impact on dispersion. In addition, terrain effects could be important. This technique could be much more useful when more complex dispersion models are coupled to a receptor model. We plan to explore this option in future work.

Note that one can also interpret the source apportionment factors as the total model error. If one had perfect knowledge of dispersion characteristics, meteorological conditions, source emissions information, and receptor data, one would expect the apportionment factors to be all ones. So the difference from one indicates the total uncertainty in the modeling and monitoring process. This implies that such a coupled model could also be used to estimate uncertainty for models.

ACKNOWLEGEMENTS – Part of this work was supported by in house funds from the director of the PSU Applied Research Laboratory. George Young, John Wyngaard, and Joel Peltier provided helpful discussions. Preliminary work was done at Utah State University with assistance from David Omer and Jesse Warrick. Thanks are due to Randy Martin for information on the Cache Valley, UT emissions inventory.

REFERENCES

Beychok, M.R., 1994, *Fundamentals of Stack Gas Dispersion,* 3rd Ed, Milton Beychok, pub., Irvine, CA, 193 pp.

Cartwright, H.M, and S.P. Harris, 1993, "Analysis of the distribution of airborne pollution using GAs," *Atmos. Environ.*, **27A**, No. 12, pp. 1783-1791.

Haupt, R.L. and S.E. Haupt, 2004, *Practical Genetic Algorithms, 2nd Edition with CD*, John Wiley & Sons, New York, NY, 255 pp.

Haupt, S.E., 2004, Genetic Algorithms and their Applications in Environmental Sciences, in *Progress of Artificial Intelligence in Sustainability Science*, J. Kropp, ed., Nova Science Publish, Inc. N.Y., in press.

Loughlin, D.H., S.R. Ranjithan, J.W. Baugh, Jr., and E.D. Brill, Jr., 2000, "Application of GAs for the Design of Ozone Control Strategies," *J. Air & Waste Manage. Assoc.*,

50, 1050-1063.

Miller, S.L., M.J. Anderson, E.P. Daly, J.B. Milford, 2002, “Source apportionment of exposures to volatile organic compounds. I. Evaluation of receptor models using simulated exposure data,” *Atmos. Env.,* **36**, 3629-3641.

Qin, R. and K. Oduyemi, 2003, Atmospheric aerosol source identification and estimates of source contributions to air pollution in Dundee, UK, *Atmos. Env.,* **37**, 1799-1809.

Venkatram, A. and J.C. Wyngaard, Ed, 1988, *Lectures on Air Pollution Modeling*, American Meteorological Society, Noston, MA. 390 pp.

Wyngaard, J.C., 1992, Atmospheric turbulence. *Annual Review of Fluid Mechanics*, **24**, 205-233.

Wyngaard, J.C. and L.J. Peltier, 1996, Experimental micrometeorology in an era of turbulence simulation, *Bound. Layer Meteor.*, **78**, 71-86.