CONTRACT NO. A932-091
FINAL REPORT
MAY 1993

An Investigation of Error Propagation in the California Air Resources Board Air Quality Model

CALIFORNIA ENVIRONMENTAL PROTECTION AGENCY

AIR RESOURCES BOARD
Research Division

AN INVESTIGATION OF ERROR PROPAGATION IN THE CALIFORNIA AIR RESOURCES BOARD AIR QUALITY MODEL

Final Report
Contract No. A932-091

Prepared for:

California Air Resources Board
Research Division
2020 L Street
Sacramento, California 95814

Prepared by:

M. Talat Odman, Menner A. Tatang, Naresh Kumar, Laurie A. McNair, Gregory J. McRae+ and Armistead G. Russell *

+6 Partridge Lane
Winchester, Massachusetts 01890

and

*11 Warriors Road
Pittsburgh, Pennsylvania 15205

MAY 1993

TABLE OF CONTENTS

ABSTRACT

An extensive series of tests have been performed to quantify the error and uncertainty propagation in the CALGRID model and its modules. Further tests have been performed on calculating the sensitivity of aspects of the calculations (e.g. chemical kinetics, horizontal transport). Also, coding checks have been conducted both computationally and manually. Recommendations of specific changes have been made, and implemented, to improve and understand model performance (e.g. the chemical ODE solver, the use of a filter in the transport algorithm, vertical transport, etc.). Many of the tests performed are unique to this project. For one, effort has been expended to test system modules within the CALGRID modeling framework when ever feasible. In addition, some new methods to test error and uncertainty propagation have been applied. Also, new tests have been conducted on specific modules as applied to an actual simulation.

It was found that the horizontal transport algorithm used is a source of significant error when concentration gradients are high. This was tested in three ways. First, the standard rotating puff test, with solid body rotation was used. This analysis, and looking at the basic formulation of one-dimensional operators applied to that test, suggested that a more severe test was desirable. An extension of that test was developed and applied to the CALGRID model. A final, very telling, test was to compare three different transport algorithms as applied to the Southern California Air Quality Study (SCAQS) data. Each of these tests suggested that the error arising from the horizontal transport solution could be of the order of 20 to 40%.

Similarly, an *in situ* test, along with more standard tests and formal sensitivity analysis techniques, was used to quantify the effect of the choice of chemical kinetics solvers was tested. In general, after modifications to the original scheme used in CALGRID, the solvers provided much less error than the transport algorithm chosen.

A unique contribution of this research is the introduction of some new computational procedures for assessing error propagation. These methods, making use of stochastic finite elements, show considerable promise as a way to quantitatively follow the effects of parameter errors. Using the vertical transport code used in CALGRID, it was found that vertical diffusivity errors and uncertainties were more significant than dry deposition and chemical conversion errors.

In summary, CALGRID can be an effective air quality model, with errors typical of most of the photochemical models currently in use. The horizontal transport algorithm is likely the largest source of error and uncertainty propagation, and the use of the non-linear filter will likely over-diffuse the emissions from some point sources. On the other hand, the transport algorithm is less diffusive than others currently in use. The code itself is relatively portable, though some non-standard FORTRAN statements were found. CALGRID should provide acceptable performance on most computational platforms, in terms of algorithm accuracy and efficiency, for use in typical photochemical air quality model applications.

Acknowledgments: The California Air Resources Board Research Division, its chief Dr. John Holmes and Mr. Bart Croes have contributed immeasurably to the success of this venture. Dr. Kit Wagner and Mr. Paul Allen of the Technical Services Division, have, as usual, contributed significantly to the progress of this project. This personal support as well as the financial support for this research, provided by the California Air Resources Board under Contract Number A932-091[1], is greatly appreciated.

[1]**Disclaimer:** A requirement of these contracts is that the following disclaimer be inserted. "The statements and conclusions of this report are those of the Contractor and not necessarily those of the State Air Resources Board. The mention of commercial products, their source, or their use in connection with material reported herein is not to be construed as either an actual or implied endorsement of such products."

LIST OF KEY SYMBOLS[2]

Symbol

$\mathbf{c}$	- Concentration vector = $(c_1,...,c_n)$
E	- Emissions of species
f	- Nonlinear right hand side vector for numerical solution
F	- Heat flux
$\mathbf{F}$	- Nonlinear operator equation
H	- Height of the model domain
H_m	- Hermite polynomial
$\mathbf{J}$	- System Jacobian
k	- Reaction rate constant
$\mathbf{k}$	- Vector of parameters = $(k_1,...,k_m)$
$\mathbf{K}$	- Turbulent eddy diffusivity tensor or cumulant for moments
K_{zz}	- Vertical component of the eddy diffusivity tensor
L	- Transport operator
L_k	- Kolmogorov length scale
L_d	- Diffusion length scale
$\mathbf{M}$	- Matrix of coefficients associated with a particular numerical scheme
M_i	- Initial mass of material in a vertical column
N_D	- Damkohler number (molecular scale)
p	- Pressure
P	- Probability density function
Q	- Heat flux
r_f	- Forward reaction rate
R	- Overall species reaction rate
RH	- Relative humidity
$\mathbf{S}$	- Matrix of coefficients or sensitivity matrix
T	- Temperature
T_k	- Kolmogorov time scale
t	- Time
$\mathbf{u}$	- Velocity vector = (u,v,w) or state variables in sensitivity studies
v_g	- Species deposition velocity
$\mathbf{V}$	- Covariance matrix
$\mathbf{x}$	- Poistion vector = (x,y,z)
Z_i	- Mixing height

Special Symbols

ρ	- Density
σ	- Standard deviation
Δz	- Mesh Spacing
ω	- Angular velocity
$<>$	- Ensemble average brackets
$[]$	- Concentration

[2]Most symbols are defined after their first use in the text. The variables used in the list are some of the more commonly used terms.

1. Introduction

Photochemical air pollution or, as it is more commonly known, smog, is an environmental problem that is both pervasive and difficult to control. An important element of any approach directed at attempting to improve the situation is a reliable means for predicting the air quality impacts of alternative emissions control measures. While many different methods have been developed, the most comprehensive and technically defensible approach has been to use mathematical models that describe, in detail, the physical and chemical processes responsible for oxidant production in the atmosphere. Typically these models are based on the principle of conservation of mass expressed in the form of differential equations that describe the concentration dynamics of both inert and chemically reactive pollutants. Figure 1.1 illustrates how the model forms an integral component of the air quality planning process.

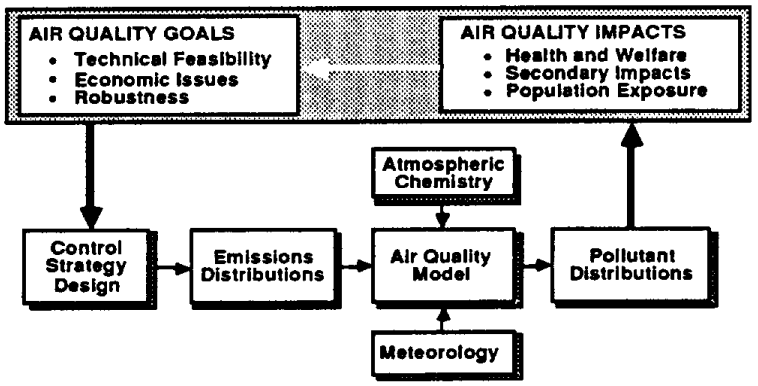


Figure 1.1 Schematic representation of the elements of the air quality planning process and the role of mathematical models.

If the models are to be of value in the air quality planning process, they need to account for the key processes occurring over the airshed of interest, including emissions, transport, turbulent diffusion, chemical reaction and removal, etc. An inevitable consequence of using models, however, is that approximations are involved and these in turn lead to uncertainties and the possiblity of errors in predictions. A critical question currently facing air pollution agencies that use the models to develop control strategies is: What are the effects of uncertainties and errors on predicted levels of control? From a practical point of view, what is needed is a clear understanding of how errors arise and how they propagate through the modeling system and, ultimately, how they affect the results of various types of calculations.

The uncertainties or sources of error may take one of five possible forms:

Physical Processes	Incompleteness in the conceptual description of the physical processes.
Structural	Errors or uncertainties in the mathematical description of the processes occurring in the atmosphere.
Algorithmic	Inaccuracies or errors that arise in the numerical solution of the governing equations.
Computational	Uncertainties that may arise from the use of different computer systems and/or operating systems.[1]
Data Related	Errors that arise from the data used by the model. The data may be specific to the airshed of interest, for example emissions and meteorological information, or it may be in parameters incorporated into the model itself (for example kinetic rate constants, stability parameters, etc.).
Human Related	Errors that arise from the misapplication of the model to conditions other than those implicit in the formulation of the model.

In some cases the distinction between error and uncertainty is not as simple as it might seem. For example, even with exact data and a correct formulation of the model, errors can arise because it is possible to represent only grid cell averages in numerical solution procedures. Some other dimensions of the problems are illustrated in Figure 1.2 where it can be seen that the goal is to determine the magnitude of the uncertainties and errors in model predictions and how they compare against the errors in observed air quality measurements. One desirable result, and the focus of much of Chapter 4, is to identify the major sources of errors in such a way as to provide information about how to correct them. The basic goal of this research is to develop meaningful tests to assess the magnitude of the errors generated by airshed models. This information can then be used to interpret model performance and in particular give insight about the range of likely discrimination between alternative emissions control strategies. While this report is

[1] With modern 32- and 64-bit computer systems, adherence to IEEE standards for representation of floating point numbers and the use of FORTRAN77 leads to differences in predicted concentration peaks that are typically much less than 0.1%. The differences, when they do occur, usually arise from different compilers and the level of code optimization.

focussed on errors in the model predictions, it is important to point out that equal attention needs to be given to assessing the effects of errors in the measurements themselves. Without a measure of the uncertainty in the air quality data, it is not possible to develop statistically meaningful comparisons between predictions and observations.

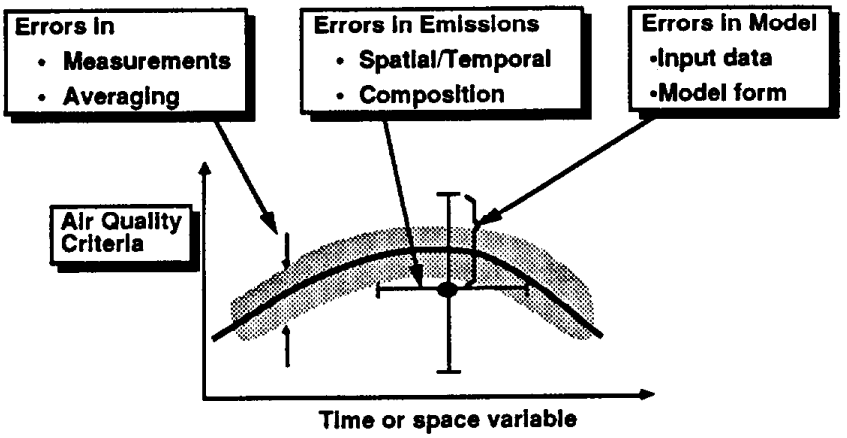


Figure 1.2: Comparison between a model prediction and observational data

In summary the key question is: What are the magnitude of the errors in the model predictions? As might be expected, there are no simple answers to this question. However, it is worthwhile to look at how the models are used in practice, and how their predictive performance can be tested. There are two critical steps in building some confidence in the performance of the model; one is to assess the level of accuracy of the basic formulation and the other is to characterize the effects of and errors uncertainties in input data. This report presents the results of how errors arise and propagate through a comprehensive photochemical airshed model that was developed for the California Air Resources Board (CALGRID) (Yamartino et al. 1989, 1992; Scire et al. 1989). While the methodologies have been tested using the CALGRID model, it is important to note that the procedures are applicable to other models used in air quality planning.

1.1 Research Tasks

In the original proposal to the California Air resources Board six tasks were proposed and they included:

Task 1: An initial project meeting to finalize the work statement with CARB staff. (This meeting was held in Sacramento on May 10, 1990.)

Task 2: Software evaluation: Several approaches to the problem were to be considered: coding level checks, module evaluation tests and data independent system integrity checks.

Task 3: Review and development of techniques to characterize the likely range of errors in the model formulation, numerical solution procedures and computational implementation. Associated with this task is the identification of key variables (for example the number of computational layers, rate constants, turbulence parameterizations, emissions, deposition velocities, etc) and their associated uncertainties.

Task 4: Application of sensitivity and uncertainty procedures to the CALGRID model.

Task 5: Model performance evaluation using SCAQS data sets.

Task 6: Preparation of the final report

During the module testing phase particular attention was given to characterizing the accuracy of the numerical integration procedures. Two steps were involved. The first was a formal error analysis to ascertain the orders of the schemes and to characterize the truncation errors. This information is needed to evaluate the performance of the modules and their ability to maintain the following properties: conservation, causality, reversibility, positivity and accuracy. Many of these techniques are described in Oran and Boris (1987) where particular attention is given to reactive transport problems. Each of the core numerical solution routines was tested, where possible, against analytic solutions or against very accurate methods. For example the chemical solver was checked against a very robust implementation of the Gear integration routine (the code used was LSODE; Hindmarsh, 1980). In evaluating the transport schemes, particular attention was given to characterizing the truncation error as a function of mesh size.

A unique feature of the research discussed in this report is the use of system level integrity checks:. One way to evaluate the performance of the system as a whole is to devise a set of tests that are independent of the quality of the input data. For example the underlying convective diffusion equation is based the principle of conservation of mass. By using emissions inventories of known composition and turning off the chemistry it is possible to assess the quality of the conservation property.

In addition to the tasks outlined in the original proposal several additional studies were carried out. For example, a detailed study was carried out to assess the validity of the K-Theory of turbulent transport in a reacting fluid. Several computational experiments were conducted to evaluate the resolution needed to resolve vertical concentration gradients.

model. The additional work was carried out to ensure that the CALGRID model provided an essentially valid description of the physical and chemical processes occurring in the atmosphere. These steps were necessary in our view because of the importance of airshed models in the air quality planning process.

Quite apart from the error analysis of the CALGRID model itself the research has resulted in several unique innovations. A suite of new test problems have been developed to evaluate numerical solution procedures. Data independent tests have been devised to evaluate the system as a whole. This study is also the first to use the Southern California Air Quality Study (SCAQS) data sets as a way to evaluate different numerical algorithms. Several new procedures for combined error and uncertainty analysis have also been developed. The new algorithms are many orders of magnitude more computationally efficient than existing techniques. During the course of the project a new version of CALGRID became available to the principal investigators. Because of the importance of the project we repeated many of the error evaluations. The extra effort was very worthwhile for some of the tests. For example, the revisions to the Quasi Steady State Approximation (QSSA) numerical procedure, used to solve the chemical kinetics, were tested and found to be significantly better than the original version described in Yamartino et al. (1989).

1.2 Structure of the Report

The subsequent sections of this report are organized into seven major sections:

- An evaluation of the mathematical formulation of CALGRID and its computer implementation (Section 2).
- A detailed discussion of the issues associated with model evaluation and the use of sensitivity and uncertainty analysis procedures to study error propagation (Section 3)
- Evaluation of the key modules that form the CALGRID model: horizontal and vertical transport, solution of the chemical kinetics, and finally coupled chemistry and transport (Section 4)
- Development and application of a new sampling methodology for uncertainty propagation through the atmospheric diffusion equation. (Section 5)
- Development and application of a direct stochastic approach to solution of the atmospheric diffusion equation. (Section 6)
- Application of CALGRID and Component Testing Using Southern California Air Quality Study (SCAQS) Data. (Section 7)

- Application of CALGRID and Component Testing Using Southern California Air Quality Study (SCAQS) Data (Section 7).
- Summary and conclusions (Section 8).

In addition to the body of the report two appendices, which contain more detailed discussions, have been included. The appendices are:

Appendix A: A comparison of fast chemical kinetic solvers for air quality modeling (*Atmospheric Environment,* in press).

Appendix B: Use of Sensitivity Analysis to Compare Chemical Mechanisms for Air Quality Modeling, (*Environmental Science and Technology*, in press).

The detailed diagnostics and analysis of the CALGRID FORTRAN code are too voluminous to reproduce in this report and has been supplied as an accompanying set of computer disks. A companion report, Kumar et al. (1992) describes the incorporation of the Carbon-Bond 4 reaction mechanism into the CALGRID model.

2. The CALGRID Model Evaluation

This chapter has been designed to introduce the CALGRID mesoscale model for photochemical air pollution, to evaluate several of the key assumptions used in its formulation and to assess the associated FORTRAN computer code. At the outset it is important to point out that the model described in Yamartino et al. (1989, 1992) and Scire et al. (1989) does represent the state-of-the-art for air quality models at the time of its formulation both in terms of its parameterizations of the physical processes and its computational implementation. The CALGRID model was a result of a project initiated by the California Air Resources Board in 1987 to upgrade and modernize the Urban Airshed Model (UAM), and is a success in that endeavor.

2.1 The CALGRID Mathematical Formulation

The CALGRID model is based on the species conservation equation for chemically reacting flows in the atmosphere. Neglecting the effects of molecular diffusion the species continuity equation is of the form

$$\frac{\partial c_i}{\partial t} + \nabla \cdot (\mathbf{u}\, c_i) = R_i[c_1 \cdots c_n;\, T,t] \tag{2.1.1}$$

where $c_i(\mathbf{x},t)$ is the concentration of species $i = 1,2,\ldots,n$, $\mathbf{u}(\mathbf{x},t)$ is the three-dimensional velocity field, R_i is the net rate of chemical reaction, T the temperature at t time and position $\mathbf{x} = \{x,y,z\}$. A key assumption in developing the CALGRID model is that (2.1.1) can be decoupled from the equations that describe the dynamics of the bulk fluid. These equations are of the form:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\mathbf{u}\, \rho) = 0 \tag{2.1.2}$$

$$\frac{\partial \rho \mathbf{u}}{\partial t} + \nabla \cdot (\mathbf{u}\mathbf{u}\, \rho) + 2\, \Omega \times \mathbf{u} = -\nabla p - \rho \mathbf{g} + \mu \nabla^2 \mathbf{u} \tag{2.1.3}$$

$$\frac{\partial \rho E}{\partial t} + \nabla \cdot (\mathbf{u} E \rho + F^H - k \nabla T) = Q_H - p \nabla \cdot \mathbf{u} \tag{2.1.4}$$

The equation for conservation of momentum (2.1.3) is an expression of Newton's second law and incorporates the effects of Coriolis forces, pressure gradients, buoyancy and viscous dissipation. While there are several ways to express energy conservation, a common form is (2.1.3) where the key terms are the radiative exchange with the surroundings F^H, heat exchange by molecular conduction and heating due to phase

change Q_H. Closure of the model system is obtained by specifying the equations of state, and the appropriate set of initial and boundary conditions.

In practice the species equation (2.1.1) can be decoupled from the momentum and energy equations (2.1.2-2.1.4). The validity of this assumption has been assessed in McRae et al. (1981) where it is shown that, under typical conditions, the pollutant species are present at such low concentration levels that they do not influence the dynamics of the carrier fluid which in this case is air. The fact that the air flow dynamics can be decoupled from the species conservation results in a major simplification. The velocity, temperature, humidity and pressure fields can be generated separately from the species continuity equation. There are a variety of ways of producing the fields and they are described in McRae et al. (1981) and Yamartino et al. (1992) and summarized in Figure 2.1.1.

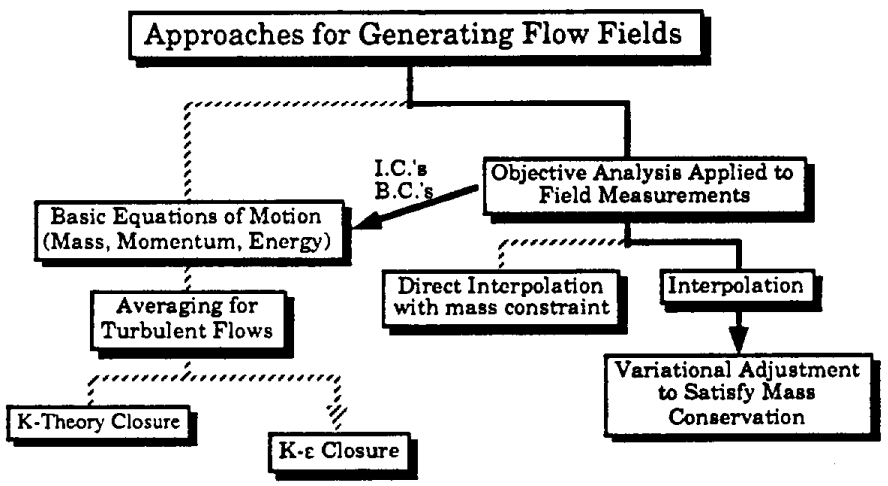


Figure 2.1.1 Summary of Procedures Currently Used to Generate Flow Fields for Air Quality Modeling

The only requirement in developing the velocity fields for use in the CALGRID model is that the flows are divergence-free, i.e.,

$$\nabla \cdot \mathbf{u} = 0 \tag{2.1.5}$$

Unlike boundary layer flows encountered in other disciplines, the location of the mixed layer is often known from atmospheric measurements. Typically the elevation of the mixed layer $Z_i(x,t)$ is determined *a priori* by interpolation of observed mixing height data.

In the atmosphere the fluid flows are typically highly turbulent and as a result the reacting species are advected by a turbulent velocity field that is usually characterized in terms of a mean and fluctuating quantity, i.e.,

$$\mathbf{u} = \bar{\mathbf{u}} + \mathbf{u}' \tag{2.1.6}$$

The fluctuating component $\mathbf{u}'$ represent those time scales of velocity variations that are typically shorter than the smallest time step. The fluctuating velocity field also induces turbulent variations in the concentrations of the form

$$[c] = [\bar{c}] + [c'] \tag{2.1.7}$$

resulting in the classical turbulence closure problem. If (2.1.6) and (2.1.7) are substituted into (2.1.1) and the Reynolds averaging rules are applied the resulting system of equations is of the form

$$\frac{\partial \langle c_i \rangle}{\partial t} + \nabla \cdot (\bar{\mathbf{u}} \langle c_i \rangle) = - \nabla \cdot \langle \mathbf{u}' c_i' \rangle + \langle R_i[c_1 \cdots c_n; T,t] \rangle \tag{2.1.8}$$

The emergence of terms like $\langle \mathbf{u}' c_i' \rangle$ results in the classical turbulence closure problem. In (2.1.8) the brackets < > denote ensemble averages. In the CALGRID model a K-theory model is used to close the equations. In this approach the turbulent transport model is assumed to be analogous to the simplest molecular models in which the flux is assumed to be proportional to the linear mean gradients. The model for a non isotropic flow is of the form

$$\langle \mathbf{u}' c_i \rangle = - \mathbf{K} \cdot \nabla \langle c_i \rangle \tag{2.1.9}$$

where now $\mathbf{K}$ is a second rank eddy diffusivity tensor that is usually assumed to be of diagonal form. Since Yamartino et al. (1992) describe the parameterization of the terms of $\mathbf{K}$ the details the formulation will not be repeated here.

One of the key assumptions used in the CALGRID model is that the term $\langle R_i[c_1 \cdots c_n; T,t] \rangle$ in (2.1.8) can be approximated by ensemble average reaction rate that is equivalent to the rate based on the ensemble species concentrations. i.e.

$$\langle R_i[c_1 \cdots c_n] \rangle = R_i[\langle c_1 \rangle \cdots \langle c_n \rangle] \tag{2.1.10}$$

The basis for the validity of this assumption is discussed in Section 2.2. Accepting for the moment (2.1.10) the final model is the so-called atmospheric diffusion equation

$$\frac{\partial \langle c_i \rangle}{\partial t} + \nabla \cdot (\bar{\mathbf{u}} \langle c_i \rangle) = \nabla \cdot \mathbf{K} \nabla \langle c_i \rangle + R_i[\langle c_1 \rangle \cdots \langle c_n \rangle; T,t] \tag{2.1.11}$$

If the ensemble averaging brackets are dropped the resulting form is the basic model used in the CALGRID model, i.e.,

$$\frac{\partial c_i}{\partial t} + \nabla \cdot (\bar{\mathbf{u}} c_i - \mathbf{K} \cdot \nabla c_i) = R_i[c_1 \cdots c_n; T, t] \tag{2.1.12}$$

The validity of this model for typical urban scale flows is discussed in considerable detail in McRae et al. (1981) and will not be repeated here. Two issues however deserve more detailed discussion. One is the assumption (2.1.10) and the other is the question of the number of vertical computational cells needed to provide adequate resolution of the concentration gradients. Both topics are discussed in the next sections.

2.2 Interaction between Diffusion and Chemical Reaction

One of the key assumptions used in formulating the CALGRID model is that K-theory provides an adequate description of the turbulent transport occurring in the atmosphere. Since many of the reaction steps in the photochemical mechanism are very fast relative to the mixing times it is important to assess the relative importance of the interplay between the fluid turbulence and the chemistry. Turbulence modeling itself is a very difficult subject let alone addressing the effects of chemistry. Georgopoulos and Seinfeld (1986) provide a comprehensive review of the subject. The discussion below is intended to show how the various mixing processes affect the chemistry.

With regard to the specific question about whether reactions are rate or diffusion limited, it is useful to introduce some more precise definitions into the discussion. Consider a bimolecular reaction of the form:

$$A + B \xrightarrow{k} \text{Products} \tag{2.2.1}$$

where k is the reaction rate constant. If the concentrations of A and B are denoted by [A] and [B], then the forward rate for (2.2.1) is given by:

$$r_f = k\,[A]\,[B] \tag{2.2.2}$$

As described in the previous section the reacting species are advected by a turbulent velocity field that is usually characterized in terms of a mean and fluctuating quantity i.e.

$$\mathbf{u} = \bar{\mathbf{u}} + \mathbf{u}' \tag{2.2.3}$$

The fluctuating component $\mathbf{u}'$ represent those time scales of velocity variations that are typically shorter than the smallest time step. The fluctuating velocity field also induces turbulent variations in the concentrations of the form

$$[A] = [\bar{A}] + [A'] \tag{2.2.4}$$

Where now $[\overline{A}]$ is the mean concentration and $[A']$ is the fluctuating component. When the Reynolds averaging rules are applied to the forward rate (2.2.2) the resulting turbulent form of the reaction rate is:

$$r_f^T = k\,[\overline{A}]\,[\overline{B}] + k\,\overline{[A'][B']} \tag{2.2.5}$$

The appearance of the term $k\,\overline{[A'][B']}$ presents the classic closure approximation. In most atmospheric modeling studies a key assumption that is made is:

$$[\overline{A}]\,[\overline{B}] \gg \overline{[A'][B']} \tag{2.2.6}$$

so that the mean concentrations can then be used to determine the overall reaction rates. The key question, of course, is under what circumstances is (2.2.6) true, and is the bulk reaction diffusionally or kinetically limited?

The relative importance of diffusion and chemistry can be expressed in terms of the dimensionless Damkohler Number N_D:

$$N_D = \frac{\text{Dispersion Time Scale}}{\text{Reaction Time Scale}} = \frac{k([A]+[B])}{2\,D} L_d^2 \tag{2.2.7}$$

where L_d is a measure of length scale across the reaction front and D is the molecular diffusivity of the species in air. When $N_D \gg 1$ (corresponding to very fast chemistry) the characteristic time for chemical reaction is short compared to that for mixing, and the phenomenon is governed not by the reaction kinetics but by the rate at which reactants are brought together at the molecular level by dispersive processes. When $N_D \ll 1$ (corresponding to slow chemistry) concentration fluctuations are dissipated before they can effect the kinetics. One important point to note is that if one of the species is in great abundance, or well mixed, then the reaction is kinetically limited.

There are two key points to keep in mind for atmospheric flows. One is that the nature of the turbulence is related to the fact that production and dissipation are not happening at the same scales. Buoyancy is feeding only the largest scales and dissipation is acting on only the smallest scales. The second key point is that typical photochemical reaction mechanism have many steps, and conclusions based on looking at individual steps in isolation can be very misleading. We will return to this point later.

To give some sense of the order of magnitudes of the relevant scales in the atmosphere consider first the smallest turbulent scales:

$$L_k = \left(\frac{\nu^3}{\varepsilon}\right)^{\frac{1}{4}} \approx 2 \text{ mm} \tag{2.2.8}$$

$$T_k = \left(\frac{\nu}{\varepsilon}\right)^{\frac{1}{2}} \approx 0.3 \text{ seconds} \tag{2.2.9}$$

where L_k and T_k are known as the Kolmogorov length and time scales, respectively, and are expressed in terms of the dissipation rate per unit mass ε and the kinematic viscosity ε (see Tennekes, and Lumley, 1972). A measure of the time scale for micromixing using typical atmospheric values is:

$$T_{Micromixing} = \frac{L_k^2}{D} \approx 0.4 \text{ seconds} \tag{2.2.10}$$

By comparison the time scales for turbulent transport by buoyancy and mechanical generation is

$$T_{Turbulent\ Diffusion} \approx O\left(\frac{Z_i^2}{K_{zz}}\right) \approx 5 - 60 \text{ minutes} \tag{2.2.11}$$

The basis for this diffusion time is developed in McRae et al. 1981, pp. 175-188 where its is shown how these rates can be derived from SF_6 tracer studies). Similarly, the time scales for source injection and mixing are O(10 minutes). The latter estimate is based on a typical source strength of ~2 ppm-m/min, a concentration level of 1 ppm and a length scale of 25 m i.e.

$$T_{Source\ Injection} \approx \frac{L_c}{E} \approx 12 \text{ minutes} \tag{2.2.12}$$

The next question is whether the molecular diffusion is rapid within the turbulent eddies compared to the mixing time of the eddies themselves? This time scale is given by

$$T_{Molecular\ Diffusion} = \frac{L_d^2}{D} = \frac{2 \times 10^{-4}\text{m}}{2 \times 10^{-5}\text{m}^2/\text{s}} \approx 2 \times 10^{-3} \text{ seconds} \tag{2.2.13}$$

which is quite fast. In other words, the concentrations are very uniform within the small eddies, and there are no distinct molecular gradients within the eddies. The key implication of the time scale analysis is that:

$$T_{Source\ Injection}, T_{Turbulent\ Diffusion}\ O(5\text{-}60 \text{ minutes}) >> T_{Micromixing}\ O(0.4 \text{ seconds}) \tag{2.2.14}$$

and

$$T_{Micromixing}\ O(0.4 \text{ seconds}) >> T_{Molecular\ Diffusion}\ O(0.002 \text{ seconds}) \quad (2.2.15)$$

In other words the reacting species are all very well mixed at the microscale (both the molecular levels and the small eddy regimes) relative to the macroscale and there is no reaction front of the type that is often seen in flame combustion. Thus the approximate length scale in the Damkohler Number calculation is the measure of molecular spacing or the mean free path. Since in air the mean free path is $O(8 \times 10^{-8}$ m) we have $N_D << 1$.. The one remaining issue is the influence of fast chemistry.

In order to illustrate some of the issues consider the following simple mechanism involving the reactions of NO, NO_2 and HCHO. When used in an atmospheric dispersion model the reaction steps might be of the form

$$NO_2 + h\nu \xrightarrow{k_1} NO + O(^3P) \quad (2.2.16)$$

$$O(^3P) + O_2 + M \xrightarrow{k_2} O_3 + M \quad (2.2.17)$$

$$NO + O_3 \xrightarrow{k_3} NO_2 + O_2 \quad (2.2.18)$$

$$HCHO + h\nu \xrightarrow{k_{4a}} 2\,HO_2 + CO \quad (2.2.19)$$

$$HCHO + h\nu \xrightarrow{k_{4b}} H_2 + CO \quad (2.2.20)$$

$$HCHO + OH \xrightarrow{k_5} HO_2 + CO + H_2O \quad (2.2.21)$$

$$HO_2 + NO \xrightarrow{k_6} NO_2 + OH \quad (2.2.22)$$

$$NO_2 + OH \xrightarrow{k_7} HNO_3 \quad (2.2.23)$$

If the mechanism is examined closely it might seem that stoichiometric balance is violated, for example, in (2.2.19). The photolysis step is in fact a lumped reaction that is derived from the combination of two elementary steps involving atomic hydrogen (H) and the formyl radical (HCO):

$$HCHO + h\nu \xrightarrow{k_{4a}} HCO + H \quad (2.2.24)$$

$$HCO + O_2 \xrightarrow{k_{4a'}} HO_2 + CO \quad (2.2.25)$$

$$HO + O_2 \xrightarrow{k_{4a''}} HO_2 \quad (2.2.26)$$

The key point to note about these reaction is that both HCO and H react extremely quickly with molecular oxygen (O_2) Using typical concentration values and reaction rate constants it is easy to see that $N_D << 1$ and so it would seem that the chemistry is kinetically limited.

The next question is the impact of the turbulence on the largest scales of motion, i.e., is (2.2.6) violated or not? Consider the forward rate of (2.2.25)

$$r_f^T = k\,\overline{[HCO]}\,\overline{[O_2]} + k\,\overline{[HCO]'\,[O_2]'} \qquad (2.2.27)$$

In the lower atmosphere the concentration of oxygen is approximately 210,000 ppm and to a first approximation it is also uniform throughout the mixed layer. Since the perturbation $[O_2]'$ from (2.2.25) is O(2 x 10^{-6} ppm) the fluctuations $[O_2]'$ relative to the mean is negligible so that the rate is in fact given by

$$r_f^T = k\,\overline{[HCO]}\,\overline{[O_2]} \qquad (2.2.28)$$

In other words while (2.2.25) and (2.2.26) are formally bi molecular they can in practice be treated as is they were unimolecular because the oxygen concentration is constant. Most of the fast radical production chemistry involves reactions with molecular oxygen.

There are however several species and reaction steps that do not directly involve reactions with molecular oxygen. Consider for example, two of the key reactions (2.2.18) and (2.2.22). Shown in Table 2.2.1 are the reaction Damkohler numbers for some typical atmospheric conditions.

Table 2.2.1

Damkohler Numbers of Two Critical Reactions

Reaction	Rate Constant (ppm units)	[A]	[B]	N_D
$NO + O_3$	23.9	0.045	0.069	3.33
$NO + HO_2$	12000.0	0.045	1.56×10^{-6}	662.

At first glance it would seem that the reaction $NO + HO_2$ is dominated by very fast chemistry. Without a more detailed understanding of the mechanism as a whole it is easy to reach the wrong conclusion that gradient transport theories are likely to be inappropriate. If we look at the simple mechanism it is a straightforward task to derive the steady state expressions for the two key radicals, hydroxyl (OH) and hydroperoxyl (HO_2):

$$[OH]_{ss} = \frac{2\,k_{4a}[HCHO]}{k_7[NO_2]} \qquad (2.2.29)$$

and

$$[HO_2]_{ss} = 2\, k_{4a} \frac{[HCHO]}{k_6[NO]} \left(1 + \frac{k_5[HCHO]}{k_7[NO_2}\right) \tag{2.2.29}$$

Both of these steady state expressions establish themselves in about 0.1 seconds which is slow compared to the mixing times at the molecular level and fast compared to the large scale transport times. The implication of these expressions are quite important because they show that the concentration of the radicals are in fact set by the time scales for reactions of the slowly reacting species. For example the time scale for HCHO photolysis is O(200 minutes). The net effect of the slow reactions of the primary species is a reduction of the overall reaction rate of steps like (2.2.22).

In summary, all of the reactions that are used in atmospheric diffusion modeling are not rate limited by either microscale turbulent diffusion or molecular diffusion. The concentration dynamics of those species that have very high kinetic rates are in almost all cases controlled by more slowly reacting species. In the air shed model the grid spacing has been chosen to reflect the gradients of these species. As a result there is no need to consider the incorporation of second order turbulent correlations in atmospheric reaction chemistry. The only exceptions are when some of the reactions take place in regions of high concentration gradients near stack exits and even in these cases the corrections to the net rates are negligible. (See Georgopoulos and Seinfeld; 1986).

2.3 Vertical Resolution of Computational Mesh

In the case of the horizontal resolution adopted by the CALGRID model it is possible to determine the minimum grid resolution in terms of the Lagrangian time scale of the turbulence. (See for example Tennekes and Lumley, 1972). As was shown in the previous section there is a complex interplay between the mixing and the chemical reaction rates. The critical question is of course is: How many layers are needed to resolve the vertical concentration profiles? There are several ways to address the problem. The simplest is just to solve the model with varying number of layers and look for the asymptotic limit where adding additional cells no longer produces a changes in predicted concentrations. Consider a Lagrangian version of the CALGRID model. The concentration dynamics of species c_i for an airshed of height H (see McRae et al. 1981 for further details of such a formulation) is given by:

$$\frac{\partial c_i}{\partial t} = \frac{\partial}{\partial z} K_{zz} \frac{\partial c_i}{\partial z} + R_i[c_1, \cdots, c_n; T, t] \tag{2.3.1}$$

Where as before R_i is the chemical reaction term and K_{zz} the vertical diffusivity. The lower level boundary conditions accounting for emissions E_i and surface removal by dry a deposition velocity v^i_g are given by:

$$v^i_g c_i - K_{zz}\frac{\partial c_i}{\partial z} = E_i[x,t] \quad ; z = 0 \tag{2.3.2}$$

At the top of the airshed, well above the mixed layer Z_i, the boundary condition corresponds to the zero flux condition:

$$K_{zz}\frac{\partial c_i}{\partial z} = 0 \quad ; z = H \tag{2.3.3}$$

Closure of the system of equations requires the specification of the initial concentration profile:

$$c_i(z,0) = c_i^0 \tag{2.3.4}$$

Since under most practical situations the systems of equations are nonlinear, a numerical solution procedure must be used and the resulting set of ordinary differential equations, after discretization, are of the form

$$M\frac{dC_i}{dt} + S\ C_i = f_i[C_1,\cdots,C_n;\ t] \tag{2.3.5}$$

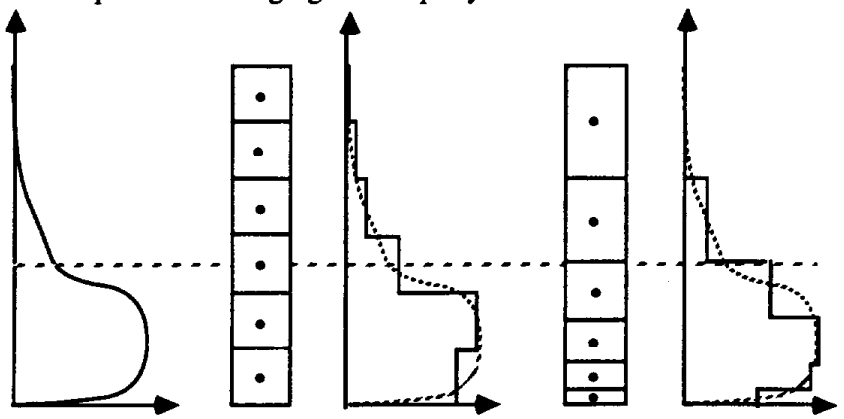
where now the vector C_i represents the concentration of species i at each of the n_z computational mesh points in the vertical direction. If the forcing term f_i is a function of time only then the analytic solution of (2.3.5) is given by

$$C_i(t) = \exp\left[-M^{-1}S\ t\right] + \int_0^t \exp\left[-(t-\tau)\,M^{-1}S\right] S^{-1} f_i(\tau)\,d\tau \tag{2.3.6}$$

or if we assume first order chemistry, fixed diffusivities and constant deposition velocities then the predictions can be compared against the analytic solution (see for example Carslaw and Jaeger, 1986; Nawrocki and Papa, 1963). Using the analytic solutions it is possible to bound the error in the numerical solution process. For the full nonlinear system with chemistry, the solution procedures embedded in CALGRID (an exponential solver and a hybrid integration technique) can be compared against highly accurate (but expensive) stiff solvers based on Gear's method. Such comparisons are included in Chapter 5 of this report.

Given an accurate solution procedure the next step is to see what happens if the number of layers is changed. Figure 2.3.1 shows two ways to approximate the vertical concentration profile for a fixed number of layers. In case (b) the mesh spacing in fixed.

Case (c) is more useful in practice because the resolution is greatest in the regions where the concentration profile is changing most rapidly.



(a) (b) (c)

Figure 2.3.1 Vertical Concentration Profile (a), A Fixed Mesh (b) and Variable Mesh (c)

Considerable care must be exercised when studies of this form are carried out to ensure that the same mass of material is present in the column of cells at the start of the integration. If M_i^o is the initial mass then the vertical mass distribution for each case must satisfy

$$M_i^0 \equiv \int_0^H C_i(z,0)\,dz = \sum_{i=1}^{n_z} C_i(\Delta z_i,0)\Delta z_i \qquad (2.3.7)$$

where n_z is the number of vertical layers and

$$\sum_{i=1}^{n_z} \Delta z_i,0) = H \qquad (2.3.8)$$

The results of such a calculation for a 24-hour simulation are shown in Figure 2.3.2. In this case the initial profile was uniform up to the top of the mixed layer and at background level from $Z_i \leq z \leq H$.

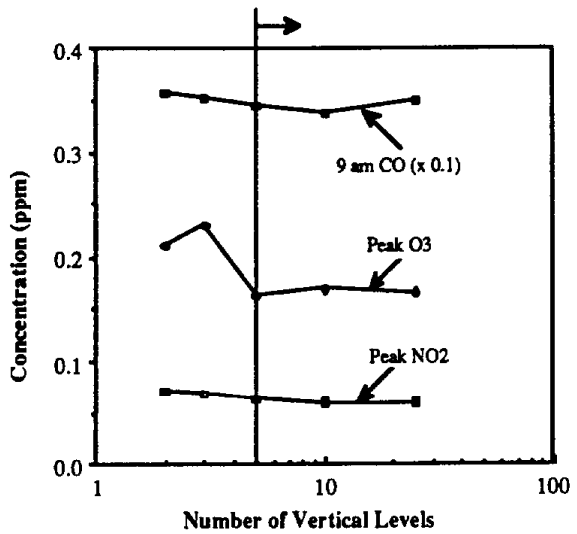


Figure 2.3.2 Predicted Surface Concentrations as a Function of the Number of Vertical Layers

Table 2.3.1 summarizes the statistics for the predicted peak concentrations for the species CO, NO_2 and O_3 where it can clearly be seen that the incremental change in surface concentrations after 5 cells is quite small. This table is a typical example of the results for the photochemical models, such as CALGRID.

Table 2.3.1

Predicted Surface Concentrations as a Function of the Number of Computational Cells used in the Vertical Direction
[Model based on (2.3.1)]

Number of Layers	CO (9 am)	Peak O_3	Peak NO_2
2	3.56	0.210	0.070
3	3.52	0.230	0.067
5	3.46	0.162	0.062
10	3.38	0.168	0.060
25	3.50	0.165	0.059

The point of the calculations presented in this section is that if the number of vertical cells is five or greater then the concentration gradients can be accurately resolved. Under strong convective mixing conditions of the type encountered in the middle of the day the vertical variation of the concentrations below the mixed layer is quite small. At night the diffusivities are so low that in effect the individual layers are uncoupled.

2.4 Computer Implementation of the CALGRID Model

The source code, input files and the output from a test case for the CALGRID model were received from the California Air Resources Board. The names of the FORTRAN files on the distribution tape are shown in Table 2.4.1 together with the INCLUDE files shown in Table 2.4.2. Further details of the function of each of the modules are contained in the User's Manual (Scire et al. 1989). The code has been subjected to a number of stringent tests, and apart from a few minor violations of the ANSI FORTRAN77 standards it is well written and easy to follow. The authors of the CALGRID code have made considerable efforts to include sufficient comments in each module so that the variable names, and processes occurring inside each module are quite clear.

Table 2.4.1

Names of FORTRAN Codes that define the CALGRID Model

BLDUP.f	RDHDEM4.f	diffvs.f	output.f	setcom.f
BLOCKDATA1.f	RDMOD.f	dow.f	partit.f	setdec.f
CHMRXN.f	RDPRM.f	dryi.f	pres3d.f	setout.f
CONSTR.f	RDTIEM2.f	fin.f	pressz.f	setup.f
DIFUN.f	RDTIEM3.f	grday.f	prise.f	setvar.f
EMATRIX.f	RESPHK.f	hadvi.f	qssa.f	svbc.f
EMQA.f	SAVPHK.f	horiz.f	rdbcon.f	text.f
ERF.f	SOLPHK.f	iaddrs.f	rdhdlbc.f	timeav.f
INTEGR.f	VRTPHK.f	iarlen.f	rdhdtbc.f	under0.f
MAIN.f	WATCON.f	incr.f	rdi1d.f	units.f
NEWPHK.f	abrdirec.f	initar.f	rdi2d.f	vadvi.f
NEWRK.f	allcap.f	inpqa.f	rdicon.f	vdcomp.f
OPENEM1.f	altonu.f	julday.f	rdirec.f	vdp.f
PHKINI.f	arrpar.f	kh2.f	rdlbc.f	vdp1.f
QCHECK.f	chapvs.f	kz.f	rdmet.f	windgrd.f
QGRID.f	chem.f	makez4.f	rdr1d.f	wrdat.f
RDEM1.f	chemi.f	makezf.f	rdr2d.f	wrout1.f
RDEM2.f	comp.f	makzeq.f	rdtbc.f	wrt.f
RDEM3.f	compz.f	met1.f	rdtcon.f	wrt2.f
RDEM4.f	datetm.f	metqa.f	rdvd.f	zjump.f
RDEMREC.f	deblnk.f	openfl.f	readcf.f	zmatrix.f
RDHDEM1.f	deltt.f	openot.f	readin.f	zshufl.f
RDHDEM2.f	depvel.f	opsplt.f	relhum.f	ztrans.f
RDHDEM3.f	diff.f	out.f	rinicon.f	

Table 2.4.2

INCLUDE Files used in the CALGRID FORTRAN Source Code

datehr.h	em1.h	gen.h	modlspc.h	vertflg.h
difcom.h	em2.h	grid.h	outpt.h	
drydep.h	em3.h	lbc.h	params.h	
drygas.h	em4.h	master.h	qa.h	
drypart.h	flags.h	methd.h	tbc.h	

The CALGRID model was compiled and executed on three different operating systems, VAX VMS, DEC Unix and Cray UNICOS. The Unix make file used to execute the code is shown in Table 2.4.3.

Table 2.4.3

Unix Make File used to Compile the CALGRID Model

```
#
#   Standard UNIX Makefile to compile, link and create an executable file for the CALGRID model.
#               Created by Gregory J. McRae and Naresh Kumar.
#
IBF = -lF77 -lI77 -lU77
#
#   Define the list of subroutines that need to be loaded
#
LIST = allcap.o altonu.o arrpar.o BLDUP.o chapvs.o chem.o chemi.o CHMRXN.o
comp.o compz.o CONSTR.o datetm.o deblnk.o deltt.o depvel.o diff.o diffvs.o
DIFUN.o dow.o dryi.o EMATRIX.o EMQA.o ERF.o fin.o grday.o hadvi.o horiz.o
iaddrs.o iarlen.o incr.o initar.o inpqa.o INTEGR.o julday.o kh2.o kz.o MAIN.o
makez4.o makezf.omakzeq.o met1.ometqa.o NEWPHK.o NEWRK.o OPENEM1.o openfl.o
openot.o opsplt.o out.o output.o partit.o PHKINI.o pres3d.o pressz.o prise.o
QCHECK.o QGRID.o qssa.o rdbcon.o RDEM1.o RDEM2.o RDEM3.o RDEM4.o RDEMREC.o
RDHDEM1.o RDHDEM2.o RDHDEM3.o RDHDEM4.o  rdhdlbc.o rdhdtbc.o rdi1d.o rdi2d.o
rdicon.o rdirec.o rinicon.o abrdirec.o rdlbc.o rdmet.o RDMOD.o RDPRM.o rdr1d.o
rdr2d.o rdtbc.o rdtcon.o RDTIEM2.o RDTIEM3.o rdvd.o readcf.o readin.o relhum.o
RESPHK.o SAVPHK.o setcom.o setdec.o setout.o setup.o setvar.o SOLPHK.o svbc.o
timeav.o under0.o units.o vadvi.o vdcomp.o vdp.o vdp1.o VRTPHK.o WATCON.o
windgrd.o wrdat.owrout1.o wrt.o wrt2.o zjump.o zmatrix.o zshuf1.o ztrans.o

.f.o: ; f77 -c -g -o $@ $*.f

calgrid.exe: $(LIST)
        f77  $(LIST) $(LIBF) -g -o calgrid.exe

#
#   Define the dependencies for each of the routines on the parameter files
#
allcap.o        : params.h
arrpar.o        : params.h master.h
MAIN.o  : params.h qa.h flags.h master.h gen.h grid.h em1.h em2.h
  em3.h vertflg.h drydep.h drygas.h drypart.h difcom.h outpt.h flags.h
chapvs.o        : params.h
chem.o          : params.h modlspc.h
chemi.o         : modlspc.h
CHMRXN.o        : modlspc.h
comp.o          : params.h gen.h grid.h outpt.h methd.h datehr.h master.h
flags.h
compz.o         : params.h gen.h grid.h
deblnk.o        : params.h
depvel.o        : params.h drydep.h gen.h grid.h methd.h
diff.o          : params.h grid.h difcom.h
dryi.o          : params.h drygas.h drypart.h drydep.h gen.h
EMQA.o          : params.h gen.h grid.h em1.h em2.h em3.h em4.h
fin.o           : params.h datehr.h qa.h
grday.o         : params.h
horiz.o         : params.h vertflg.h
iaddrs.o        : params.h master.h
iarlen.o        : params.h master.h
```

```
incr.o          : params.h
inpqa.o         : params.h outpt.h gen.h grid.h vertflg.h
julday.o        : params.h
makez4.o        : params.h
makzeq.o        : params.h vertflg.h
met1.o          : params.h methd.h
metqa.o         : params.h grid.h methd.h
NEWPHK.o        : modlspc.h
NEWRK.o         : modlspc.h
OPENEM1.o       : params.h gen.h
openfl.o        : params.h
openot.o        : params.h
opsplt.o        : params.h master.h grid.h gen.h drydep.h outpt.h datehr.h
flags.h
out.o           : params.h datehr.h
output.o        : params.h grid.h gen.h outpt.h drydep.h datehr.h
partit.o        : params.h gen.h grid.h outpt.h em1.h em2.h em3.h methd.h
master.h
PHKINI.o        : modlspc.h
QGRID.o         : params.h gen.h grid.h datehr.h em1.h em2.h em3.h em4.h
rdbcon.o        : params.h gen.h grid.h lbc.h
RDEM1.o         : params.h em1.h
RDHDEM1.o       : params.h em1.h
RDHDEM2.o       : params.h em2.h
RDHDEM3.o       : params.h em3.h
RDHDEM4.o       : params.h em4.h
rdhdlbc.o       : params.h gen.h grid.h lbc.h
rdhdtbc.o       : params.h gen.h grid.h tbc.h
rdicon.o        : params.h gen.h grid.h
rdlbc.o         : params.h gen.h grid.h
rdmet.o         : params.h methd.h
RDMOD.o         : modlspc.h
RDPRM.o         : modlspc.h
rdtcon.o        : params.h gen.h grid.h
rdtbc.o         : params.h gen.h grid.h
rdvd.o          : params.h gen.h drydep.h
readcf.o        : params.h difcom.h drygas.h drypart.h drydep.h em4.h flags.h
gen.h grid.h methd.h outpt.h qa.h vertflg.h
readin.o        : params.h
RESPHK.o        : modlspc.h
rinicon.o       : params.h master.h grid.h gen.h
SAVPHK.o        : modlspc.h
setcom.o        : params.h master.h grid.h
setout.o        : params.h outpt.h
setdec.o        : modlspc.h
setup.o         : params.h gen.h grid.h outpt.h em2.h em3.h drydep.h methd.h
master.h qa.h flags.h
SOLPHK.o        : modlspc.h
svbc.o          : params.h gen.h grid.h
timeav.o        : params.h grid.h gen.h outpt.h
vadvi.o         : params.h vertflg.h
vdcomp.o        : params.h drygas.h drypart.h
vdp.o           : params.h
vdp1.o          : params.h
VRTPHK.o        : modlspc.h
windgrd.o       : params.h
wrout1.o        : params.h qa.h gen.h grid.h methd.h outpt.h
zjump.o         : params.h vertflg.h
zshuf1.o        : params.h
ztrans.o        : params.h vertflg.h
```

The particular machines used to evaluate the CALGRID code were a DEC 3100, a DEC Station 5000/240, a VAX/VMS 6400 and a Cray YMP/864 operating under UNICOS. In all cases when the sample problem was executed the differences in the predictions were negligible. Standard software utilities were used to compare the output files. The maximum observed concentration differences were less than 0.1%. The differences, when they arose were due to computer word size and the level of compiler optimization. For example, depending on the requested level of code optimization when using the f77 compiler on the DEC Station 5000, the predicted concentration fields were different in the last significant decimal digit.

One feature of the CALGRID model that the authors might consider changing is the use of binary format input/out files. While binary files can reduce the needed storage and the input output time the benefits are often illusory. There is frequently a need to look at the files and if they are in binary format they cannot be easily opened and interpreted with a conventional text editor. A further difficult is that binary files often cause problems when the are transmitted over telecommunications networks. A final argument for the use of formatted ASCII text files is that they avoid difficulties when the code and files are transferred between machines that have different floating point representations. Again as noted above the use of binary files is not a major problem if the code is to remain resident on a single computer system. The original version of the model described in McRae et al. (1981) used binary files, the current version now employs text format.

An additional recommendation about the code is to avoid the use of a single array to hold all the variables used by the model. On systems that have small physical memories the use of a single large array can cause an excessive number of page faults.

2.5 System and Coding Checks

Before carrying out a detailed evaluation of the CALGRID model and the formulation of the individual modules the first step after reproducing the test case on different computer systems was to examine the computer code itself. Several Computer Aided Software Environment (CASE) tools were used together with the use of a structured walkthrough technique introduced by Yourdon (1989). The initial stage involved the preparation of the data dependency graphs to make sure that the code we used was consistent with the structure presented in the User's manual (Scire et al., 1989). These figures enabled the checking for undefined data elements, undefined files, control blocks, non-referenced data elements. The tools also provide listings of module inputs

and outputs that can be checked against the CALGRID documentation. Once the data dependency graphs were developed (see White, 1989 for details) the next step was to carry out a structured walkthrough. Basically a walkthrough is a review process designed to evaluate the quality of computer software. Several approaches to the problem were considered including compliance of the code with FORTRAN77 standards, checking for portability, etc. One of the most useful tools used during this process was a public domain program called **ftnchek** that is available from the Oak Ridge National Laboratory Netlib[1] facility.

Ftnchek (short for Fortran checker) is designed to detect certain errors in a Fortran program that a compiler usually does not. **Ftnchek** is not primarily intended to detect syntax errors. Its purpose is to assist the user in finding semantic errors. Semantic errors are legal in the Fortran language but are wasteful or may cause incorrect operation. For example, variables which are never used may indicate some omission in the program; uninitialized variables contain garbage which may cause incorrect results to be calculated; and variables which are not declared may not have the intended type. Ftnchek is intended to assist users in the debugging of their Fortran program. It is not intended to catch all syntax errors. This is the function of the compiler.

Ftnchek is invoked through a command of the form:

ftnchek [-option -option ...] output=filename [filename ...]

(The brackets indicate something which is optional. The brackets themselves are not actually typed.) Here options are command-line switches or settings, which control the operation of the program and the amount of information that will be printed out. If no option is specified, the default action is to print error messages, warnings, and informational messages, but not the program listing or symbol tables. The options used to evaluate the CALGRID code were:

-declare If this flag is set, all identifiers whose datatype is not declared in each module will be listed. This flag is useful for helping to find misspelled variable names, etc. The same listing will be given if the module contains an IMPLICIT NONE statement. Default = no.

---

[1] **ftnchek** can be obtained by sending a message with a title "send ftnchek from fortran" to the Internet address: "netlib@ornl.gov". The code is automatically returned to the sender. All that needs to be done to implement **ftnchek** is to compile and load the source code. The user system must have a C compiler. The documentation described above is derived directly from the downloaded file FTNCHEK.DOC.

-extern Causes **Ftnchek** to report whether any subprograms invoked by the program are never defined, or are multiply defined. Ordinarily, if **Ftnchek** is being run on a complete program, each subprogram other than the intrinsic functions should be defined once and only once somewhere. Turn off this switch if you just want to check a subset of files which form part of a larger complete program, or to check all at once a number of unrelated files which might each contain an unnamed main program. Subprogram arguments will still be checked for correctness. Default = yes.

-f77 This flag catches language extensions which violate the Fortran 77 standard. Such extensions may cause the program not to be portable. Examples include the use of underscores in variable names; variable names longer than six characters; statement lines longer than 72 characters; and nonstandard statements such as the DO ... ENDDO structure. **Ftnchek** does not report on the use of lowercase letters. Default=no.

-list Specifies that a listing of the Fortran program is to be printed out with line numbers. If **Ftnchek** detects an error, the error message follows the program line with a caret (^) specifying the location of the error. If no source listing was requested, **Ftnchek** will still print out any line containing an error, to aid the user in determining where the error occurred. Default = no.Print source listing of program. Default = no.

-portability **Ftnchek** will give warnings for a variety of non-portable usages. These include the use of tabs except in comments or inside strings, the use of hollerith constants, and the equivalencing of variables of different data types. This option does not produce warnings for violations of the Fortran 77 standard, which may also cause portability problems. To catch those, use the -f77 option. Default = no.

-sixchar One of the goals of the **Ftnchek** program is to help users to write portable Fortran programs. One potential source of nonportability is the use of variable names that are longer than six characters. List any variable names which clash at 6 characters length. Default = no.

-symtab A symbol table will be printed out for each module, listing all identifiers mentioned in the module. This table gives the name of each variable, its datatype, and the number of dimensions for arrays. An asterisk (*) indicates that the variable has been implicitly typed, rather than being named in an explicit type declaration statement. The table also lists all subprograms invoked by the module, all common blocks declared, etc. Default = no. Print out symbol table. Default = no.

-usage This switch is on by default. It causes **Ftnchek** to list all variables which may be used before they are initialized, or which are given a value but never subsequently used, or which are declared but never used. Sometimes **Ftnchek** makes a mistake about this. Usually it errs on the side of giving a warning where no problem exists, but in rare cases it will fail to warn where the problem does exist. See the

section on bugs for examples. If variables are equivalenced, the rule used by **Ftnchek** is that a reference to any variable implies the same reference to all variables it is equivalenced to. Default = yes.

-common=n This setting varies the strictness of checking of common blocks. Level 3 is the strictest: it requires that in each declaration of a given common block, correspondingvariables agree in data type and (if arrays) size and number of dimensions. Levels 1 and 2 require only that corresponding memory locations agree in data type. Thedifference between Levels 1 and 2 is that Level 2 warns if the blocks are not equal in total length, while Level 1 does not. Level 0 suppresses all checking. Default = 3.

The output from the **Ftnchek** program for the CALGRID model is too voluminous to reproduce here.[2] **Ftnchek** produces four main types of messages. They are portability warnings, other warnings, informational messages, and syntax errors. Portability warnings specify nonstandard usages that may not be accepted by other compilers. Other warning messages report potential errors that are not normally flagged by a compiler. Informational messages consist of warnings which may assist the user in the debugging of their Fortran program. A brief summary of the types of error message generated during the evaluation of the CALGRID code is set out below.

"Warning: file contains tabs. May not be portable." **Ftnchek** expands tabs to be equivalent to spaces up to the next column which is a multiple of 8. Some compilers treat tabs differently, and also it is possible that files sent by electronic mail will have the tabs converted to blanks in some way. Therefore files containing tabs may not be compiled correctly after being transferred. **Ftnchek** does not give this message if tabs only occur within comments or strings.

"nonstandard type usage in expression." The program contains an operation such as a logical operation between integers, which is not standard, and may not be acceptable to some compilers.

"Common block has mixed character and non-character variables", "Common block has long data type following short data type." The ANSI standard requires that if any variable in a common block is of type CHARACTER, then all other variables in the same common block must also be of type CHARACTER. Some compilers additionally require that if a common block contains mixed data types, all long types (namely DOUBLE PRECISION and COMPLEX) must precede all short types (namely integer, real, etc.). The following messages are warning messages:

"Integer quotient expr converted to real", "integer quotient expr used in exponent." The quotient of two integers results in an integer type result, in which the fractional part is dropped. If such an integer expression involving division is later converted to a real data type, it may be that a real type division had been

---

2 Since the output is over 500 pages the listing is has been stored on a computer readable disk that is available from the authors.

intended. Likewise, if it is used as an exponent, it is likely that a real type division was intended.

"real truncated to intg." **Ftnchek** has detected an assignment statement which has a real expression on the right, but an integer variable on the left. The fractional part of the real value will be lost. If you explicitly convert the real expression to integer using the INT or NINT intrinsic function, no warning will be printed. A similar message is printed if a double precision expression is assigned to a real variable, etc.

"Continuation follows comment or blank line." **Ftnchek** issues this warning message to alert the user that a continuation of a statement is interspersed with comments, making it easy to overlook.

"Declared but never referenced." Detects any identifiers that were declared in your program but were never used, either to be assigned a value or to have their value accessed. Variables in common are excluded.

"Variables used before set." This message indicates that an identifier is used to compute a value prior to its initialization. Such usage may lead to an incorrect value being computed.

"Variables may be used before set." Similar to used before set except that **Ftnchek** is not able to determine its status with certainty. **Ftnchek** assumes a variable may be used before set if the first usage of the variable occurs prior in the program text to its assignment.

"Variables set but never used" **Ftnchek** will notify the user when a variable has been assigned a value, but the variable is not otherwise used in the program. Usually this results from an oversight.

Type has been implicitly defined" **Ftnchek** will flag all identifiers that are not explicitly typed and will show the data type that was assigned through implicit typing. This provides support for users who wish to declare all variables as is required in Pascal or some other languages. This message is printed only when the -symtab option is in effect.

"Identifiers which are not unique in first six chars" Warns that two identifiers which are longer than 6 characters do not differ in first 6 characters. This is for portability: they may not be considered distinct by some compilers. This message is printed only if the -sixchar option was selected.

"Subprogram NAME: varying length argument lists:" An inconsistency has been found between the number of dummy arguments (parameters) a subprogram has and the number of actual arguments given it in an invocation. **Ftnchek** keeps track of all invocations of subprograms (CALL statements and expressions using functions) and compares them with the definitions of the subprograms elsewhere in the source code. The Fortran compiler normally does not catch this type of error.

"Subprogram NAME: argument data type mismatch at position n " The subprogram's nth actual argument (in the CALL or the usage of a function) differs in datatype from the nth dummy argument (in the SUBROUTINE or FUNCTION declaration). For instance, if the user defines a subprogram by

```
SUBROUTINE SUBA(X)
 REAL X
```

and elsewhere invokes SUBA by

```
CALL SUBA(2)
```

Ftnchek will detect the error. The reason here is that the number 2 is integer, not real. The user should have said

```
CALL SUBA(2.0)
```

When checking an argument which is a subprogram, **Ftnchek** must be able to determine whether it is a function or a subroutine. The rules used by **Ftnchek** to do this are as follows: If the subprogram, besides being passed as an actual argument, is also invoked directly elsewhere in the same module, then its type is determined by that usage. If not, then if the name of the subprogram does not appear in an explicit type declaration, it is assumed to be a subroutine; if it is explicitly typed it is taken as a function. Therefore, subroutines passed as actual arguments need only be declared by an EXTERNAL statement in the calling module, whereas functions must also be explicitly typed in order to avoid generating this error message.

"Subprogram NAME: argument usage mismatch" **Ftnchek** detects a possible conflict between the way a subprogram uses an argument and the way in which the argument is supplied to the subprogram. The conflict can be one of two types, as outlined below.

"Dummy arg is modified, Actual arg is const or expr" A dummy argument is an argument as named in a SUBROUTINE or FUNCTION statement and used within the subprogram. An actual argument is an argument as passed to a subroutine or function by the caller. **Ftnchek** is saying that a dummy argument is modified by the subprogram, i.e. its value will be changed in the calling module. The corresponding actual argument should not be a constant or expression, but rather a variable or array element which can be legitimately assigned to.

"Dummy arg used before set, Actual arg not set." Here a dummy argument may be used in the subprogram before having a value assigned to it by the subprogram. The corresponding actual argument should have a value assigned to it by the caller prior to invoking the subprogram.

"Common block NAME: varying length" A common block declared in different subprograms has different numbers of variables in it in different declarations. This is not necessarily an error, but it may indicate that a variable is missing from the list.

"Common block NAME: data type mismatch at position n" The nth variable in the common block differs in data type in two different declarations of the common block. By default (common strictness level 3), **Ftnchek** is very picky about common blocks: the variables listed in them must match exactly by data type and array dimensions. That is, the legal pair of declarations in different modules:

COMMON /COM1/ A,B

and

COMMON /COM1/ A(2)

will cause **Ftnchek** to give warnings at strictness level 3. These two declarations are legal in Fortran since they both declare two real variables. At strictness level 1 or 2, no warning would be given in this example.

Variable names may be longer than six characters. The standard specifies six as the maximum. Variable names may contain underscores, which are treated the same as alphabetic letters. The VAX version of **Ftnchek** also allows dollar signs in variable names, but not as the initial character.

2.6 Application of the forchek System to CALGRID

With the **ftnchek** system it is a straightforward task to check the FORTRAN of the CALGRID code. The actual output from **ftnchek**, including the individual program listings is over 500 pages and, for obvious reasons is not reproduced here. The complete output is available from the authors as a computer readable file CALGRID.LIST. In order to illustrate the use of **ftnchek** consider two of the CALGRID routines:

`QSSA.F` which integrates the chemical kinetic differential equations using the quasi-steady state approximation

`RDHDLBC.f` which reads the header records from an unformatted lateral boundary condition (BCON) file.

The programs were chosen at random to illustrate how **ftnchek** works. In the listings below the highlighted areas correspond to those points in the module where possible problems might occur. The first code is a key one for numerical integration of the chemical kinetics. The first potential error that is flagged by **ftnchek** is the fact that the variable 'nflc' in the argument list is not used in the code itself. The variable is probably a remnant of an earlier testing phase. The fact that the variable 'nflc' is defined but not used does not effect the correct execution of the code. The next coding problem detected occurs on lines 10-11 and 19-22 where there are embedded tab stops that are denoted by '*'. The fact that the tabs occur in comment statements ensures that there will be no compiler errors but again the point of using **Ftnchek** is to avoid problems associated with portability issues or incompatibilities between compilers.

Further down in the code at lines 41 and 57 there are examples of poor programming practice. At line 41 if the concentration is negative or less than 10^{-19} then it is reset to the floor value of 10^{-19}. A similar problem occurs on line 57 where in this case the floor value is 10^{-15}. There is no justification for this choice in the documentation manual or in the code itself. Such constants depend on the units being used, the machine precision and the desired accuracy of the solution.

```
FTNCHEK Version 2.5 January 1992

File qssa.f:

  1 C-----------------------------------------------------------------------
  2       SUBROUTINE QSSA(NSP,DT,CON,FORM,LOSS,nflc)
  3 C-----------------------------------------------------------------------
  4 C
  5 C --- CALGRID    VERSION:  1.2     LEVEL:  890531                    QSSA
  6 C
  7 C***********************************************************************
  8 C     THIS SUBROUTINE INTEGRATES THE DIFFERENTIAL EQUATIONS
  9 C     USING QUASI-STEADY-STATE APPROXIMATION METHOD
 10 C     *REF ; HESSTVEDT ET AL., INT. J. OF CHEM. KINETICS,*
 11 C     *                VOL. 10, PP. 971-994 (1978)       *
 12 C
 13 C***********************************************************************
 14 C
 15 C     ARGUMENT VARIABLES
 16 C     ------------------
 17 C
 18 C     NSP  : NUMBER OF ACTIVE SPECIES
 19 C     DT   : INTEGRATION TIME STEP                    * 0.5 MIN HERE *
 20 C     CON  : CONCNTRATION OF ACTIVE SPECIES                   * PPM *
 21 C     FORM : FORMATION TERM IN THE DIFFERENTIAL EQUATION * PPM/MIN *
 22 C     LOSS :      LOSS TERM IN THE DIFFERENTIAL EQUATION *   1/MIN *
 23 C
 24 C --- QSSA CALLED BY: CHMRXN
 25 C --- QSSA CALLS:     NONE
 26 C-----------------------------------------------------------------------
 27 C
 28       REAL UPLIM,LOLIM,CON(*),FORM(*),LOSS(*)
 29 C
 30 C     SET THE UPPER AND LOWER LIMITS WITH PREDISCRIBED DT = 0.5 MIN
 31 C
 32       PARAMETER(UPLIM =20. , LOLIM = 0.02)
 33 C
 34 C     LOCAL FUNCTION DECLARATION
 35       FUNC(AA,BB,CC,TT) = (-(- AA * CC+BB) * EXP(-AA * TT) + BB) / AA
 36         ia=0
 37         ib=0
 38         ic=0
 39 C
 40       DO 1 IR = 1 , NSP
 41         if(con(ir).lt.1e-19)con(ir)=1e-19
 42 c         if(nflc.eq.1)then
 43 c         write(88,*)ir,form(ir),loss(ir)
 44 c         endif
```

```
 45 C
 46          IF      (LOSS(IR) .LT. LOLIM) THEN
 47 C
 48 C     SIMPLE EULER FORMULA
 49 C
 50             CON(IR) = CON(IR) + (FORM(IR) - LOSS(IR) * CON(IR)) * DT
 51          ia = ia +1
 52          ELSE IF(LOSS(IR) .GT. UPLIM) THEN
 53 C
 54 C     EQUILIBRIUM FORMULA
 55 C
 56 c            write(6,*)ir,form(ir),loss(ir)
 57             if(form(ir).lt.1e-15)form(ir)=1e-15
 58             CON(IR) = FORM(IR) / LOSS(IR)
 59          ib=ib+1
 60          ELSE
 61 C
 62 C     ANALYTICAL FORMULA
 63 c            write(6,*)'loss,form,con',loss(ir),form(ir),con(ir)
 64 C
 65             CON(IR) = FUNC(LOSS(IR),FORM(IR),CON(IR),DT)
 66          ic=ic+1
 67          ENDIF
 68  1    CONTINUE
 69       RETURN
 70       END


Module QSSA: subr

External subprograms referenced:

       EXP: intrns

Statement functions defined:

      FUNC: real*

Variables:

      Name Type Dims      Name Type Dims      Name Type Dims      Name Type Dims
        AA real*            BB real*            CC real*           CON real  1
        DT real*          FORM real  1          IA intg*            IB intg*
        IC intg*            IR intg*         LOLIM real          LOSS real  1
      NFLC intg*           NSP intg*            TT real*         UPLIM real

* Variable not declared. Type has been implicitly defined.


Variables declared but never referenced in module QSSA:
      NFLC*
  * Dummy argument


Identifiers of undeclared type in module QSSA:
        AA                  BB                  CC                  DT
      FUNC                  IA                  IB                  IC
        IR                NFLC                 NSP                  TT


 Warning: file contains tabs.  May not be portable.

 0 syntax errors detected in file qssa.f
 1 warning issued in file qssa.f
```

```
Subprogram QSSA never invoked
        defined in module QSSA line 2 file qssa.f
```

The second example illustrates the bulk of the diagnostics developed by ftnchek in processing the CALGRID code. In this case the warning messages are indicative of relatively minor problems. There are four minor violations of the ANSI FORTRAN77 standard. In the case of the include statements in lines 34, 35, 36, 37 the problem is the use of lower case letters for 'include', for most modern compilers this is not an issue even though the standard require the upper case form 'INCLUDE'.

Lines 79-84 are an example of a more subtle problem. In this case some previous parts of an if test condition have been effectively removed by changing them to comment records. The warning is again a pointer to poor programming practice as it is possible to forget the need for additional statements after line 79 containing the beginning of the 'IF' test. At the end of the listing are several other warning messages. One set refers to the fact that two of the common blocks, set by using the include statements on lines 35 and 37, have a mismatch in variable type. Normally this is not a problem but on some computer systems there can be problems with half word alignment. The final set of warnings are that there are seven variables referenced in the routine (see for example lines 91-93 where the names are longer than the 6 character ANSI standard. Again this is not a problem on most compilers but there are some that only use the first six characters. In the is case the first six characters are all unique and so there is no chance of a problem.

```
FTNCHEK Version 2.5 January 1992

File rdhdlbc.f:

      1 c-----------------------------------------------------------------
      2       subroutine rdhdlbc(iunit,lprt)
      3 c-----------------------------------------------------------------
      4 c
      5 c --- CALGRID    Version:  1.4A    Level:  890630              RDHDLBC
      6 c                J. Scire, SRC
      7 c
      8 c --- PURPOSE:  Read the header records from an unformatted lateral
      9 c               boundary condition (BCON) file
     10 c
     11 c --- INPUTS:
     12 c            IUNIT - integer     - Fortran unit number of BCON file
     13 c             LPRT - logical     - Flag controlling printing of header
     14 c                                  record data (F=suppress, T=print)
     15 c
     16 c      Common Block /GEN/ variables:
     17 c            NSA, CSPEC(mxspec)
     18 c      Common Block /GRID/ variables:
```

```
     19 c             IGTYPE, IVGTYP, NX, NY, NZ, DGRID, XORIG, YORIG, IUTMZN
     20 c     Parameters:
     21 c               IO6, MXSPEC, MXNZP1
     22 c
     23 c --- OUTPUT:
     24 c
     25 c     Common Block /LBC/ variables:
     26 c             FNAMEB, IGTYPB, IVTYPB, NXB, NYB, NZB, DELXB, DELYB,
     27 c             XORIGB, YORIGB, IUTMZB, NSAB, IBDATB, IBTIMB,
     28 c             IEDATB, IETIMB, VRSB, LABELB, CSLSTB(mxspec)
     29 c
     30 c --- RDHDLBC called by: RDBCON
     31 c --- RDHDLBC calls:     none
     32 c-----------------------------------------------------------------
     33 c
     34       include 'params.h'
              ^
Warning near line 34 col 7: Nonstandard syntax
Including file params.h:
      1 c-----------------------------------------------------------------
      2 c --- PARAMETER statements                                    CALGRID
      3 c-----------------------------------------------------------------
      4 c (listing off)
     35       include 'gen.h'
              ^
Warning near line 35 col 7: Nonstandard syntax
Including file gen.h:
      1 c-----------------------------------------------------------------
      2 c --- COMMON BLOCK /GEN/ -- General run control information,    CALGRID
      3 c                           file types
      4 c-----------------------------------------------------------------
(listing off)
     36       include 'grid.h'
              ^
Warning near line 36 col 7: Nonstandard syntax
Including file grid.h:
      1 c-----------------------------------------------------------------
      2 c --- COMMON BLOCK /GRID/ -- Grid parameters                   CALGRID
      3 c-----------------------------------------------------------------
(listing off)
     37       include 'lbc.h'
              ^
Warning near line 37 col 7: Nonstandard syntax
     38 c
Including file lbc.h:
      1 c-----------------------------------------------------------------
      2 c --- COMMON BLOCK /LBC/ -- Lateral boundary condition         CALGRID
      3 c                           parameters (from unformatted
      4 c                           BCON file)
      5 c-----------------------------------------------------------------
(listing off)
     39       LOGICAL lprt
     40 c
     41 c --- Header Record #1 - Grid information, beginning & ending
     42 c                        dates/times
     43 c
     44       read(iunit)fnameb,igtypb,ivtypb,nxb,nyb,nzb,delxb,delyb,xorigb,
     45      1 yorigb,iutmzb,nsab,ibdatb,ibtimb,iedatb,ietimb,vrsb,labelb
     46 c
     47       if(fnameb.ne.'BCON'.and.fnameb.ne.'bcon')then
     48          write(io6,10)fnameb,iunit
     49 10       format(/1x,'ERROR in SUBR. RDHDLBC -- file name does not ',
     50      1   'match expected value'/
     51      2   1x,'Expected file name: BCON or bcon'/
```

```
     52        3   1x,'    File name read: ',a12/
     53        4   1x,'        Unit number: ',i4)
     54            stop
     55         endif
     56 c
     57         if(nsab.gt.mxspec)then
     58            write(io6,12)nsab,mxspec
     59 12         format(/1x,'ERROR in SUBR. RDHDLBC -- No. advected species ',
     60        1   ' in BCON file is greater than current array dimension'/1x,
     61        2   '    No. advected species in file: ',i5/1x,
     62        3   'Current Array dimension (MXSPEC): ',i5)
     63            stop
     64         endif
     65 c
     66 c --- Header Record #2 - Species list
     67         read(iunit)(cslstb(n),n=1,nsab)
     68 c
     69 c
     70 c ----------------------------------------
     71 c --- QA checks on header record variables
     72 c ----------------------------------------
     73         ierr=0
     74 c
     75 c --- Check that grid parameters are consistent with control file
     76         dgridkm=0.001*dgrid
     77         xorigkm=0.001*xorig
     78         yorigkm=0.001*yorig
     79         if((nxb.ne.nx).or.(nyb.ne.ny).or.(nzb.ne.nz).or.
     80 c       1    (abs(delxb-dgridkm).gt.0.001).or.
     81 c       2    (abs(delyb-dgridkm).gt.0.001).or.
     82        3    (abs(xorigb-xorigkm).gt.0.001).or.
               ^
Warning near line 82 col 6: Continuation follows comment or blank line
     83        4    (abs(yorigb-yorigkm).gt.0.001).or.
     84        5    (iutmzb.ne.iutmzn))then
     85            ierr=1
     86            write(io6,15)
     87 15         format(/1x,'ERROR in subr. RDHDLBC'/)
     88            write(io6,*)'Mismatch in grid parameters -- NXB = ',nxb,
     89        1   ' NX = ',nx,' NYB = ',nyb,' NY = ',ny,' NZB = ',nzb,
     90        2   ' NZ = ',nz,' DELXB = ',delxb,' DELYB = ',delyb,
     91        3   ' DGRIDKM = ',dgridkm,' XORIGB = ',xorigb,
     92        4   ' XORIGKM = ',xorigkm,' YORIGB = ',yorigb,
     93        4   ' YORIGKM = ',yorigkm,' IUTMZB = ',iutmzb,' IUTMZN = ',
     94        5   iutmzn
     95         endif
     96 c
     97         if((igtypb.ne.igtype).or.(ivtypb.ne.ivgtyp).or.(nsab.ne.nsa))then
     98           ierr=1
     99           write(io6,15)
    100           write(io6,*)'Mismatch in grid types or no. species ',
    101        1  '-- IGTYPB = ',igtypb,' IGTYPE = ',igtype,' IVTYPB = ',ivtypb,
    102        2  ' IVGTYP = ',ivgtyp,' NSAB = ',nsab,' NSA = ',nsa
    103         endif
    104 c
    105         do 20 i=1,nsa
    106         if(cslstb(i).ne.cspec(i))ierr=2
    107 20      continue
    108         if(ierr.eq.2)then
    109            write(io6,15)
    110            write(io6,*)'Species names and ordering must match that used ',
    111        1   'in the model'
    112            do 24 i=1,nsa
    113            write(io6,22)i,cslstb(i),cspec(i)
```

```
114 22        format(1x,'Species number: ',i3,3x,'BCON species name: ',a12,
115       1   3x,'Model species name: ',a12)
116 24        continue
117         endif
118 c
119 c --- Terminate the run if any errors encountered
120         if(ierr.ge.1)stop
121 c
122 c --- WRITE CONTENTS OF HEADER RECORDS (if requested)
123         if(lprt)then
124            write(io6,102)
125 102        format(///1x,13('----------')//1x,'Header record data from ',
126       1    'the unformatted BCON boundary condition file')
127            write(io6,103)fnameb,igtypb,nxb,nyb,delxb,delyb,xorigb,
128       1    yorigb,iutmzb,nsab,ibdatb,ibtimb,iedatb,ietimb,vrsb,labelb
129 103        format(/1x,'FNAMEB: ',a12/1x,'IGTYPB: ',i6/1x,'NXB:    ',i6/
130       1    1x,'NYB:    ',i6/1x,'DELXB:  ',f10.3/1x,'DELYB:  ',f10.3/
131       2    1x,'XORIGB: ',f10.3/1x,'YORIGB: ',f10.3/1x,'IUTMZB: ',i6/
132       1    1x,'NSAB:   ',i6/1x,'IBDATB: ',i6/1x,'IBTIMB: ',i6/
133       2    1x,'IEDATB: ',i6/1x,'IETIMB: ',i6/
134       3    1x,'VRSB:   ',a12/1x,'LABELB: ',a12)
135            write(io6,105)
136 105        format(4x,'SPECIES'/)
137            do 110 i=1,nsab
138            write(io6,106)cslstb(i)
139 106        format(1x,a12)
140 110        continue
141         endif
142 c
143         return
144         end
```

Module RDHDLBC: subr

External subprograms referenced:

ABS: intrns

Common blocks referenced:

GEN GRID LBC

Common block GEN line 8 module RDHDLBC has mixed character and non-character variables (may not be portable)
Common block LBC line 9 module RDHDLBC has mixed character and non-character variables (may not be portable)
Variables:

Name	Type	Dims	Name	Type	Dims	Name	Type	Dims	Name	Type	Dims
AREAM2	real*		CSLSTB	char	1	CSPEC	char	1	DELXB	real*	
DELYB	real*		DGRID	real*		DGRIDKM	real*		DZMIN	real*	
FNAMEB	char		I	intg*		IBDATB	intg*		IBDY	intg*	
IBHR	intg*		IBMO	intg*		IBTIMB	intg*		IBYR	intg*	
IEDATB	intg*		IEM1REC	intg*		IERR	intg*		IETIMB	intg*	
IGTYPB	intg*		IGTYPE	intg*		IO10	intg*		IO11	intg*	
IO12	intg*		IO15	intg*		IO16	intg*		IO17	intg*	
IO18	intg*		IO20	intg*		IO5	intg*		IO6	intg*	
IO7	intg*		IO8	intg*		IO9	intg*		IRLG	intg*	
ISPLST	intg*	2	ITBCON	intg*		ITEM1	intg*		ITEM2	intg*	
ITEM3	intg*		ITEM4	intg*		ITICON	intg*		ITTCON	intg*	
IUNIT	intg*		IUTMZB	intg*		IUTMZN	intg*		IVGTYP	intg*	
IVTYPB	intg*		LABELB	char		LPRT	logl		METHINT	intg*	
MX2	intg*		MX4	intg*		MX5	intg*		MXARR	intg*	
MXBTYP	intg*		MXCOL	intg*		MXINT	intg*		MXIOP	intg*	

```
    MXMAIN intg*       MXNSNZ intg*        MXNXY intg*         MXNZ intg*
     MXNZM intg*      MXNZMP1 intg*       MXNZP1 intg*       MXPDEP intg*
      MXSG intg*       MXSPEC intg*         MXSS intg*        MXVAR intg*
         N intg*          NSA intg*         NSAB intg*         NSDD intg*
       NSE intg*        NSPEC intg*       NSUBTS intg*           NX intg*
       NXB intg*         NXM1 intg*         NXM2 intg*           NY intg*
       NYB intg*         NYM1 intg*         NYM2 intg*           NZ intg*
       NZB intg*          NZL intg*         NZP1 intg*          NZU intg*
      VRSB char          XLAT real*        XLONG real*         XMOL real* 1
     XORIG real*       XORIGB real*      XORIGKM real*          XTZ real*
     YORIG real*       YORIGB real*      YORIGKM real*       ZBOTC1 real*
     ZFACE real* 1        ZTOP real*       ZTOPC1 real*

* Variable not declared. Type has been implicitly defined.


Identifiers of undeclared type in module RDHDLBC:
    AREAM2              DELXB               DELYB              DGRID
   DGRIDKM              DZMIN                   I             IBDATB
      IBDY               IBHR               IBMO             IBTIMB
      IBYR             IEDATB            IEM1REC               IERR
    IETIMB             IGTYPB             IGTYPE               IO10
      IO11               IO12               IO15               IO16
      IO17               IO18               IO20                IO5
       IO6                IO7                IO8                IO9
      IRLG             ISPLST             ITBCON              ITEM1
     ITEM2              ITEM3              ITEM4             ITICON
    ITTCON              IUNIT             IUTMZB             IUTMZN
    IVGTYP             IVTYPB            METHINT                MX2
       MX4                MX5              MXARR             MXBTYP
     MXCOL              MXINT              MXIOP             MXMAIN
    MXNSNZ              MXNXY               MXNZ              MXNZM
   MXNZMP1             MXNZP1             MXPDEP               MXSG
    MXSPEC               MXSS              MXVAR                  N
       NSA               NSAB               NSDD                NSE
      NSPEC             NSUBTS                 NX                NXB
      NXM1               NXM2                 NY                NYB
      NYM1               NYM2                 NZ                NZB
       NZL               NZP1                NZU               XLAT
     XLONG               XMOL              XORIG             XORIGB
   XORIGKM                XTZ              YORIG             YORIGB
   YORIGKM             ZBOTC1              ZFACE               ZTOP
    ZTOPC1
```

```
Names longer than 6 chars in module RDHDLBC (nonstandard):
   DGRIDKM            IEM1REC            METHINT            MXNZMP1
   RDHDLBC            XORIGKM            YORIGKM
```

```
 0 syntax errors detected in file rdhdlbc.f
 6 warnings issued in file rdhdlbc.f

Subprogram RDHDLBC never invoked
        defined in module RDHDLBC line 2 file rdhdlbc.f
```

The purpose of the two examples was to point out that there are elements of the CALGRID code that violate the ANSI FORTRAN77 standard and as a result the code may not be portable across systems. The experience of the authors of this report is that

most of the warning messages are just that -- warnings. Most modern compilers can cope with these differences and as was reported above the code has been successfully compiled and executed on several different computer systems.

2.7 Conclusions

Based on a detailed evaluation of the model formulation and its computer implementation it can be concluded that the CALGRID model is a state-of-the-science representation of the physical and chemical processes occurring in the atmosphere based on when the model was formulated and initially delivered. While advances have occurred in the knowledge of physical and chemical processes in the atmosphere, numerical algorithms (as will be discussed in proceeding chapters), and software engineering practices since that time, the current version of CALGRID should be considered a viable tool for conducting air quality modeling studies.

3. Sensitivity/Uncertainty Analysis and Error Propagation

The key conclusion from the previous section was that the mathematical formulation of the CALGRID model is an essentially correct theoretical description of the processes occurring in the atmopshere. Given the mathematical model the next step is to evaluate how errors in the data, components parts of the process parameterizations and numerical solution algorithms contribute to uncertainties in the predictions. This section provides the background needed for the detailed evaluation to be presented in Chapter 4.

Rather than working with the atmospheric diffusion equation it is useful to consider a more general system of equations of the form

$$\mathbf{F}(\mathbf{u},\mathbf{k}) = 0 \tag{3.1}$$

where **F** is a general algebraic or differential operator, for example the atmospheric diffusion equation, **u** is a vector of n output variables $\{u_1,....u_n\}$ and $\mathbf{k} = \{k_1,...,k_m\}$ a set of m parameters. The parameters of interest could be structural variables that change the dimension of the model or its numerical approximation, kinetic rate constants, deposition velocities, emissions distributions, etc. From a practical point of view it is important to distinguish between the different types of parameter errors or uncertainties. Systematic errors can occur when the parameters values are either biased high or low relative to their true value. A typical example of this type of error might be a reaction rate constant derived from the literature. If the likely range of variation can be established then it is possible to evaluate the effects of the error. Typically rate constants are reported as mean values together with estimates of their standard deviations.

In order to assess the effects of errors in the rate constants many different values must be sampled from the statistical distribution and then used in the model. The ensemble of predictions resulting from sampling parameter values from the error distribution enables the determination of the uncertainty in the predictions. As it will be shown in Chapter 6, systematic errors can induce large uncertainties in the predictions. A more subtle source of uncertainty can arise from the randomness of the data used to determine the parameter values themselves. Consider for example the species deposition velocities. As a parcel of air moves over the airshed it can encounter different surface roughnesses, material surfaces and stability conditions. As a result the deposition velocity may exhibit a seemingly random set of fluctuations.

From a practical point of view what is needed is an approach that will show the effects of these parameter uncertainties and errors on the predictions of the model. Koda et al. (1979); McRae et al. (1982); Liu et al. (1976); Tilden et al. (1981) and McRae et al. (1981) provide detailed discussions of a wide variety of procedures and the issues associated with using them on models like CALGRID. In essence there are five basic factors that need to be considered:

1. The accuracy of the basic mathematical formulation itself in describing the processes occuring in the atmosphere.
2. Given that most practical models are nonlinear and require numerical solution, what is the accuracy of the underlying algorithm?
3. What is the extent of the errors and parameter variations to be considered?
4. The combined roles of sensitivity and uncertainty analysis.
5. The computational cost of solving the model and carrying out the analysis.

From a practical point of view the dominant consideration in selecting an approriate techniques is its computational cost since a single solution of the CALGRID model for just one parameter combination make take several hours. A related issue is the amount of effort needed to implement the particular technique. Some techniques do not require extensive programming effort beyond that need to solve the model while others can require considerable additional effort on the part of the investigator.

The ultimate goal of any study of error propagation is to determine the effects of parameter changes on the predictions. Since most models require numerical solution, the outputs needed to define the response surface, or how the predictions vary as the parameters are changed, will only be available for a finite set of parameter combinations. Given this situation the basic problem then becomes how to sample the parameter space with sufficient regularity top adequately characterize $\mathbf{u}(\mathbf{k})$. An analysis which accounts for simultaneous variations in all of the parameters over their full range of uncertainties is called a global method. Conversely, local analyses attempt to infer the shape or value of the response surface at a particular point.

3.1 Sensitivity Analysis

Because of the high computational costs associated with carrying out sensitivity studies just perturbing individual parameters is not sufficient. If, however, we start with the premise that we would like to carry out such analyses it is possible to develop an entirely

different approach. Typically the equations encountered in airshed modeling, after numerical discretization, are represented in the general operator form:

$$\mathbf{F}(\mathbf{u},\mathbf{k}) \equiv \frac{d\mathbf{u}}{dt} - \mathbf{f}(\mathbf{u},\mathbf{k},t) = 0 \tag{3.2}$$

Typically there is associated with (3.2) an output operator $\mathbf{H}(\mathbf{u})$ that maps the model predictions $\mathbf{u}(\mathbf{k})$ into a form that is compatible with the actual observed variables $\mathbf{y} = \{y_1,\ldots,y_{nobs}\}$

$$\mathbf{y} = \mathbf{H}(\mathbf{u},\mathbf{k}) \tag{3.3}$$

Note that (3.3) may or may not be a linear operator. Both (3.2) and (3.3) are typically very large and quite stiff because there is a wide variation in the underlying time and space scales. In practice it is important to recognize that (3.3) may more than just the concentration at a point. The operator could describe the peak one hour concentration observed during a simulation, the population exposure or deposition flux. For any of these metrics we are interested in the sensitivity of changes in the state variables to variations in the parameters.

The sensitivity matrix is of the form:

$$\mathbf{S} \equiv s_{ij} = \left(\frac{\partial u_i}{\partial k_j}\right)_{|\mathbf{k} = \mathbf{k}^*} \tag{3.4}$$

where $\mathbf{k}^*$ is the nominal set of parameter values. In practice the partial derivatives in (3.4) can sometimes be derived analytically but usually they must be numerically approximated. Unfortunately, conventional low order finite difference approximations can cause slow convergence or search directions that lead to suboptimal solutions. High order finite difference methods require many more function evaluations. An alternate approach is to consider the adjoint system:

$$\frac{\partial}{\partial t}\left(\frac{\partial \mathbf{u}}{\partial \mathbf{k}}\right) + [\mathbf{J}_F(\mathbf{u},\mathbf{k})]\left(\frac{\partial \mathbf{u}}{\partial \mathbf{k}}\right) + \left(\frac{\partial \mathbf{F}}{\partial \mathbf{k}}\right) = 0 \tag{3.5}$$

where $\mathbf{J}_F(\mathbf{u},\mathbf{k})$ is the Frechet derivative, or best locally linear approximation, to the operator $\mathbf{F}(\mathbf{u},k)$ at the nominal parameter value $\mathbf{k}^*$. The Frechet derivative is formally defined by:

$$\lim_{\|\mathbf{h}\| \to 0} \frac{\| \mathbf{F}(\mathbf{u},\mathbf{k}+\mathbf{h}) - \mathbf{F}(\mathbf{u},\mathbf{k}) - \mathbf{J}_F(\mathbf{u},\mathbf{k})(\mathbf{h})\|}{\|\mathbf{h}\|} \to 0 \tag{3.6}$$

where $\mathbf{h}$ is a vector of perturbations to the elements of $\mathbf{u}$. For simple functions $\mathbf{J}_F(\mathbf{u},\mathbf{k})$ is the system Jacobian. Rather than solve (3.5) and (3.2) as a large set of coupled differential equations it is useful to recognize that most of the information needed to

calculate the gradients can be derived from the solution of the model itself. For example, in most numerical solution procedures the approximate solution values are calculated at $t = t_0, t_1, \ldots, t_n$ where $t_n = t_{n-1} + h_n$ for some step size h_n according to the formulae:

$$\mathbf{u}_n = \mathbf{R}\left[\frac{d\mathbf{u}_{n-j}}{dt}, \mathbf{u}_{n-j}, t_{n-j}; \mathbf{k}\right]_{j = 0,\ldots,q} \tag{3.7}$$

$$\frac{d\mathbf{u}_n}{dt} = \mathbf{S}\left[\frac{d\mathbf{u}_{n-j}}{dt}, t_{n-j}; \mathbf{k}\right]_{j = 0,\ldots,q} \tag{3.8}$$

where the subscript n indicates a numerical approximation of the vector **u** at t_n, and q is the order of the scheme. The exact functional forms of **R** and **S** and depend on the particular integration technique. The resulting system of nonlinear algebraic equations that arise when (3.7-8) are substituted into (3.2) are typically solved using Newton's method:

$$\left[\frac{\partial \mathbf{R}_n}{\partial \mathbf{u}_n} + \frac{\partial \mathbf{f}_n}{\partial \mathbf{u}_n}\right]^{(m)} \left\{\mathbf{u}_n^{(m+1)} - \mathbf{u}_n^{(m)}\right\} = -\left\{\frac{\partial \mathbf{u}_n}{\partial t} - \mathbf{f}_n(\mathbf{x}_n, \mathbf{k})\right\}^{(m)} \tag{3.9}$$

where now m is the iteration number. The cost of solving (3.9) is typically dominated by the need for repeated solutions of the linear algebraic system:

$$\mathbf{A}\,\mathbf{x} = \mathbf{b} \tag{3.10}$$

where the matrix **A** is associated with the left hand side of (3.1):

$$\mathbf{A} = \left[\frac{\partial \mathbf{R}_n}{\partial \mathbf{u}_n} + \frac{\partial \mathbf{f}_n}{\partial \mathbf{u}_n}\right] \tag{3.11}$$

If now (3.7, 3.8) are substituted into (3.5) it is possible to show that the gradients are given by:

$$\left[\frac{\partial \mathbf{R}_n}{\partial \mathbf{u}_n} + \frac{\partial \mathbf{f}_n}{\partial \mathbf{u}_n}\right]^{(m)} \left\{\frac{d\mathbf{x}_n}{dt}\right\} = -\left\{\mathbf{W}\left(\frac{\partial \mathbf{u}_n}{\partial t}, t_{n-j}, \mathbf{k}\right) + \frac{\partial \mathbf{f}_n(\mathbf{x}_n, \mathbf{k})}{\partial \mathbf{k}}\right\}^{(m)} \tag{3.12}$$

where **W** has a similar form to (3.7). Again we can express (3.12) in the same manner as (3.11):

$$\mathbf{AS} = \mathbf{D} \tag{3.13}$$

where as before:

$$[\mathbf{S}] \equiv s_{ij} = \frac{\partial u_i}{\partial k_j} \quad ; \; i=1,2,\cdots,n, \;\; j=1,2,\cdots,m \tag{3.14}$$

Since the matrix **A** is common to both systems, saving the LU decomposition after a successful convergence of (3.9) provides a way to extract the gradient from (3.13) by a simple back substitution. In addition to the obvious computational savings the gradients

have an accuracy comparable to the solution of the basic model. A computer code called SENSODE was developed that incorporates most of the above ideas and it has been used to carry out a series of sensitivity studies of the reaction mechanisms embedded within the CALGRID model Carbon Bond 4 (CB-4) and Lurmann, Carter and Coyner (LCC). (Further details of the results are contained in Appendix B).

For small variations in **k**, (3.6) is the best local linear approximation to the actual response surface. If the response of the model is highly nonlinear then (3.6) may not be a good reflection of the variation of **u**(**k**) away from the nominal parameter value **k***. Figure 3.2 illustrates the limitation of local methods when they are applied to problems which involve large uncertainties in the parameters.

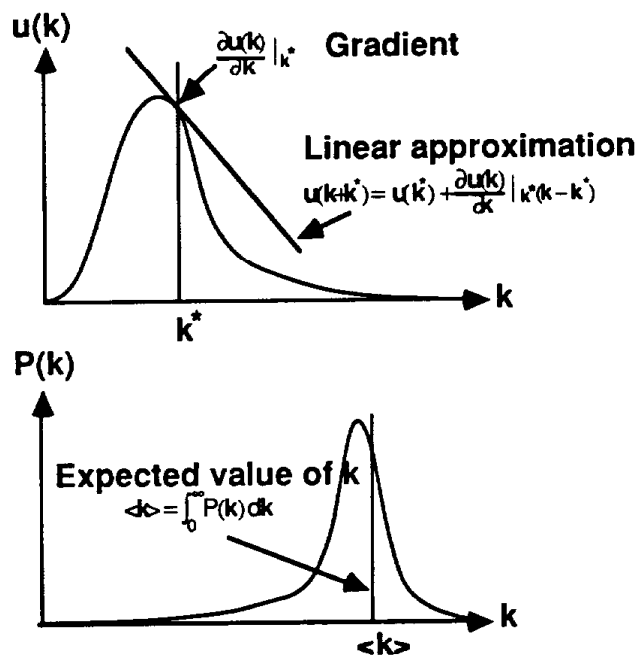


Figure 3.2 Linear and nonlinear sensitivity analysis of a model output with respect to parameters variations together with the probability density function associated with **k**.

3.2 Uncertainty Analysis

In Figure 3.2 the region of highest sensitivity occurs at a point where the likelihood of the particular value occuring in practice is small. This particular example highlights the fact that so far in the discussion all the values of **k** have been treated as being equally likely; however in practice, the parameters often have non-uniform

probability distributions. While the response surface or sensitivity is independent of all assumptions about the likely parameter combinations, the expected value or other statistical measures depend on both the sensitivity and the probability density functions for the parameters. For example the expected value of the predictions is given by

$$<u_i(\mathbf{k})> = \int \cdots \int u_i(\mathbf{k})\, P(\mathbf{k})\, dk_1 \cdots dk_m \tag{3.15}$$

where $P(\mathbf{k})$ is the multidimensional probability density function associated with the parameters. For example if the parameters were to be jointly normally distributed then $P(\mathbf{k})$ would be of the form

$$P(\mathbf{k}) = \frac{1}{(2\pi)^{\frac{n}{2}} |V|^{\frac{1}{2}}} \exp\left[- \mathbf{k}^T V^{-1} \mathbf{k}\right] \tag{3.16}$$

where $\mathbf{V}$ is the covariance matrix. Another measure of uncertainty is the variance in the predictions associated with parameter variations i.e.

$$\sigma_i^2[u_i(\mathbf{k})] = \int \cdots \int [u_i(\mathbf{k}) - <u_i(\mathbf{k})>]^2\, P(\mathbf{k})\, dk_1 \cdots dk_m \tag{3.17}$$

One of the major difficulties in carrying out an error analysis when there are statistical distributions associated with the parameter values is that the model must be solved many times in order to develop accurate solutions to the integrals of the form (3.15) and (3.17). There are a variety of procedures that can be used including Monte Carlo methods and the Fourier Amplitude Sensitivity (FAST) technique. These and other techniques are reviewed in McRae et al. (1981).

In summary, a sensitivity analysis then refers to the influence of parameter variations on the output whereas a combined sensitivity/uncertainty error analysis considers the additional factor of the probability density functions of the parameter values. Regardless of the refinements in knowledge of parameter accuracy the global sensitivity remains the same. One of the key points to take away from the discussion is that just because a parameter may be uncertain or be subject to error it does not necessarily follow that the predictions of the model have the same level of uncertainty. In fact as will be shown in Chapters 4 and 5 some of the processes are self-compensating or limited in the extent of their variation because there are other constraints acting on the system.

As an illustration of these issues consider a one-dimensional advection diffusion equation

$$\frac{\partial c}{\partial t} + \frac{\partial uc}{\partial x} = \frac{\partial}{\partial x} K_{xx} \frac{\partial c}{\partial x} \tag{3.18}$$

where K_{xx} is the diffusion coefficient and u is the velocity field u(x,t). From a mathematical standpoint (3.18) is formally a parabolic partial differential equation. Under typical atmospheric conditions with u = 5 m/s, K_{xx} = 100 m^2/s and a concentration gradient of say 1 ppm/km then it is easy to show from a scale analysis that

$$\text{Advective Flux (uc)} >> \text{Diffusive Flux} \left(K_{xx}\frac{\partial c}{\partial x}\right) \tag{3.19}$$

and so (3.18) exhibits almost hyperbolic type characteristics. In fact it is this property that causes so much difficulty in developing numerical algorithms (to be discussed in Chapters 4 and 7). If we consider an even simpler form of (3.18), where both the velocity field and the diffusion coefficient are constants, the resulting equation is of the form

$$\frac{\partial c}{\partial t} + u\frac{\partial c}{\partial x} = K\frac{\partial^2 c}{\partial x^2} \tag{3.20}$$

and it is possible to develop a very simple finite difference approximation of the form

$$\frac{c_m^{n+1} - c_m^n}{\Delta t} + u\left[\frac{c_{m+1}^n - c_{m-1}^n}{2\,\Delta x}\right] = K\left[\frac{c_{m+1}^n - 2\,c_m^n + c_{m-1}^n}{\Delta x^2}\right] \tag{3.21}$$

where m is the space index and n the time level. Equation (3.21) is the standard first-order accurate in time, second-order accurate in space discrete approximation (see Strikwerda, 1989). The overall stability limit for the scheme is

$$K\frac{\Delta t}{\Delta x^2} = K\mu \le \frac{1}{2} \tag{3.22}$$

The difference scheme (3.21) can be written in the equivalent form

$$c_m^{n+1} = (1 - 2\,K\mu)c_m^n + K\mu(1 - \alpha)\,c_{m+1}^n + K\mu(1+\alpha)c_{m-1}^n \tag{3.23}$$

where

$$\mu = \frac{\Delta t}{\Delta x^2} \qquad \text{and} \qquad \alpha = \frac{\Delta x\,u}{2\,K} = \frac{u\,\lambda}{2\,K\,\mu} \tag{3.24}$$

Provided that the stability condition (3.22) is satisfied it is possible to show that for the parabolic system (3.20) that the maximum value of |c(x,t)| will not increase as t increases (see Strikwerda, 1989)

$$\sup_x |c(x,t)| \le \sup_x |c(x,t')| \quad \text{if } t > t' \tag{3.25}$$

The finite difference scheme (3.23) will have the same property if and only if

$$\alpha = \frac{\Delta x\,u}{2\,K} \le 1 \tag{3.26}$$

That is if the condition (3.26) is satisfied as well as the stability condition then for the finite difference scheme

$$|c_m^{n+1}| = (1 - 2\,K\mu)|c_m^n| + K\mu(\,1 - \alpha)\,|c_{m+1}^n| + K\mu(1+\alpha)|c_{m-1}^n| \le \max_m |c_m^n| \qquad (3.27)$$

we have

$$\max_m |c_m^{n+1}| \le \max_m |c_m^n| \qquad (3.28)$$

On the surface the result (3.28) is very attractive since it would suggest that there would be no overshoot, the solutions would not exhibit oscillatory behavior and the numerial approximation is consistent with the original model (3.20). The basic problem is that for typical horizontal atmospheric flows (u ~ 5 m/s and K ~ 100 m^2/s) and typical grid cell spacing (Δx ~ 5000 m) we have that

$$\alpha = \frac{\Delta x\, u}{2\,K} = \frac{5000 \times 5}{2 \times 100} = 125 >> 1 \qquad (3.29)$$

Under these conditions it is possible to show that the solutions will be oscillatory. For example Figure 3.3 presents the results of the transport of a concentration profile using the same conditions as above with a time step Δt of 600 seconds. From an inspection of the plot it is quite clear that there is more than 30% overshoot in the peak and the solution profile is oscillatory.

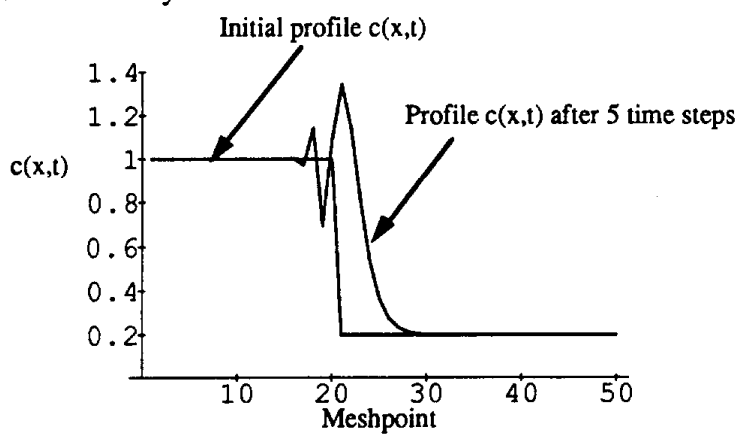


Figure 3.3 Predicted concentration profile after five time steps using the difference scheme (3.21)

With the difference scheme (3.21) the only way to avoid the oscillations is to reduce the mesh size to less than 40 m, since from (3.26) we require that

$$\Delta x \le \frac{2\,K}{u} = \frac{2 \times 100}{5} = 40 \text{ m} \qquad (3.30)$$

which in turn involves a considerable increase in computational cost and storage requirements. In order to understand the origin of the problem it is useful to regognize that term 2K/u in (3.30) corresponds to the cell Reynolds number in fluid flow or the Peclet Number in heat flow problems. The condition (3.30) is a requirement for the difference scheme to behave qualitatively like the original parabolic partial differential equation. It is important to recognize that (3.30) is <u>not</u> a stability requirement, since stability only deals with the limits as Δx and Δt tend to zero, (3.30) is always satisfied for Δx small enough. In fact for the conditions used in the sample problem the finite difference scheme is operating well below the CFL limit (See Roache, 1976). The oscillations that occur when (3.30) is violated are not the result of instability. They do not grow excessively; they are only the result of inadequate spatial resolution.

One way of avoiding the restriction (3.30) is to use upwind differencing of the convection term.

$$\frac{c_m^{n+1} - c_m^n}{\Delta t} + u\left[\frac{c_m^n - c_{m-1}^n}{\Delta x}\right] = K\left[\frac{c_{m+1}^n - 2\,c_m^n + c_{m-1}^n}{\Delta x^2}\right] \tag{3.31}$$

or

$$c_m^{n+1} = [1 - 2\,K\mu(1+\alpha)]c_m^n + K\mu\, c_{m+1}^n + K\mu(1+2\alpha)c_{m-1}^n \tag{3.32}$$

If the term $1 - 2K\mu(1 + \alpha)$ is positive (it is 0.4 for the sample problem), then (3.32) satisfies the condition (3. 28) and there will be no oscillations. The result has however been achieved by introducing another problem. As can be seen in Figure 3.4 there is a considerable diffusion of the initial profile beyond what might be expected from the physical processes occuring in the atmosphere.

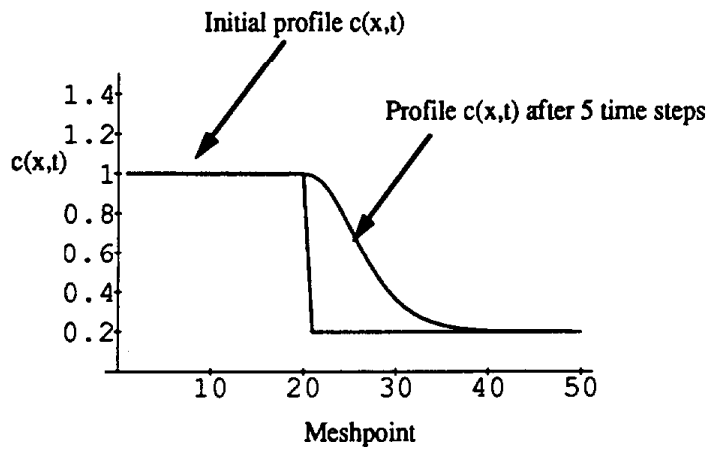


Figure 3.4 Predicted concentration profile after five time steps using the difference scheme (3.32)

The origin of the enhanced diffusion can be seen by rewriting (3.31) in the equivalent form:

$$\frac{c_m^{n+1} - c_m^n}{\Delta t} + u\left[\frac{c_{m+1}^n - c_{m-1}^n}{2\,\Delta x}\right] = (K + \frac{u\,\Delta x}{2})\left[\frac{c_{m+1}^n - 2\,c_m^n + c_{m-1}^n}{\Delta x^2}\right] \qquad (3.33)$$

In other words the effective diffusion coefficient is now K + u $\Delta x/2$. Again for the sample problem this corresponds to a numerical diffusion coefficient of 100 + 12,500 m^2/s. The artificial diffusion coefficient is two orders of magnitude greater than the turbulent diffusivity. This result illustrates an important point about uncertainty analysis. Even if there were to be a factor of 10 uncertainty in the diffusion coefficient its effect on the predictions of numerical schemes of the form (3.33) would be neglible because of the low order accuracy of the numerical approximation.

3.3 Conclusions

The key conclusion to be drawn from this section is the need to be careful in carrying out an error propagation analysis and in particular to understand the different sources of errors that can arise in air quality modeling. For example, depending on the numerical solution scheme the errors in parameter values for the diffusion coefficient are neglible in comparision to the errors that might arise from a poor choice of an integration scheme. While the numerical schemes used in the CALGRID model are considerably more sophisticated and accurate than the simple form (3.33) the basic ideas are still relevant. An important caution that can be derived from this section is that just because

there might be a large error associated with a particular parameter value it does not always follow that there will be similarly large errors in the prediction.

4. Module Evaluation

When mathematical models are used to describe various physical and chemical processes of atmospheric air pollution, there are always some discrepancies between observations and model prediction. These discrepancies are due to a variety of factors, including approximation in the model formulation, uncertainties in the model inputs, and representativeness of the measurements. Model predictions are compared against observed air quality. Determining the sources of errors is not an easy task. Since the processes involved are highly non-linear, it is almost impossible to pinpoint the sources of the errors just by looking at the simulation results.

The effects of model formulation, numerical solution procedures and computer implementation play an important role in the outcome of simulations. The sources of errors in the photochemical airshed model developed for the California Air Resources Board, CALGRID (Yamartino et al., 1992), are being investigated. Model components are tested to determine the significance of errors introduced. Level checks are conducted to assess how these errors are propagated and how they contribute to the final results. This report describes the method used in the component error and error propagation analysis and presents some preliminary results.

4.1 Background

Before describing our approach, it is important to provide some background information about the CALGRID airshed model. CALGRID is an Eulerian grid model based on the atmospheric diffusion equation. The operator splitting technique is used for solving this equation. The solution is marched in time as follows:

$$c_{t+\Delta t} = (L_x L_y L_z R L_z L_y L_x)\, c_t \tag{4.1}$$

where L_x and L_y are one-dimensional horizontal transport, L_z is the vertical transport and R is the chemical reaction operator. These operators constitute the major components of CALGRID.

4.2 Methodology

Two approaches were adopted for the assessment of uncertainties in the routines for horizontal transport, vertical and chemistry transport and their effect on model

predictions. The first approach is breaking the code into individual modules and studying each one separately. However, CALGRID makes use of a single vector to store most relevant variables. Therefore, breaking the code and analyzing each module individually would require redesign of this data structure. Also, with this approach, it would be impossible to evaluate the code's integrity and consistency. The performance of several modules together could not be assessed and a true picture of the program's modularity could not be drawn.

The second approach is to maintain the structure of the program and perform tests that suppress the desired components either via input data or by minor changes in the code. CALGRID is equipped with a sufficient number of diagnostic parameters. By setting these parameters appropriately or by changing their values internally, it is possible to test the model's components individually and in combination. This approach was found more viable in the assessment of the overall performance. The steps taken in performing the tests will be described in detail so that the tests can be repeated or the way they were performed can be open to discussion.

As mentioned above, the major components of CALGRID are the horizontal transport, chemistry and vertical transport. In the rest of this report, each module will be tested individually or in combination with the others and the errors involved in each module will be determined.

4.3 Horizontal Transport

CALGRID splits the horizontal transport into two one-dimensional operators and uses the Chapeau function scheme for the solution. One-dimensional operators lead to tridiagonal systems of equations that are solved very efficiently. However, operator splitting is associated with loss of accuracy. This issue will be discussed in a special test problem. The one-dimensional Chapeau function scheme can not approximate smooth solutions near sharp gradients. Accurate, monotonic solutions can only be assured by means of non-linear mechanisms. CALGRID is equipped with several filtering mechanisms to eliminate the ripples produced by the Chapeau scheme. First, a two-pass Forester filter is used. The mass conservative Forester filter is very effective in eliminating local noise waves. The local property of the filter maintains resolvable features such as peak pollutant concentrations. However, convergence to strictly monotonic solutions slow down considerably after the first few iterations. That is why the Forester filter is applied only twice and then the donor cell mass borrowing technique

is used. If there are still some negative concentrations in the field after the application of the donor cell filter, these concentrations are simply set equal to zero. This last procedure is not mass conservative but is not a major error source either.

4.4 Rotating Puff Test

The advection of a pollutant puff in a rotational velocity field is a standard problem for testing horizontal transport algorithms. It presents a good test case for identifying diffusion, dispersion and phase errors. However, it is important to note that a rigid body rotation field yields constant x velocity along any line parallel to the x-axis, and, likewise, a constant y component of velocity along any line parallel to the y-axis. (Of note, by extension, given any straight line, the velocity component in the direction of that line is constant.) For schemes that split horizontal transport into two one-dimensional operators, the problem becomes constant velocity advection. This is not the most severe test case but was included in our analysis because of its popularity.

The rotational velocity field is defined as

$$u_r = 0$$
$$u_\theta = \omega r \tag{4.4.1}$$

where ω is constant. In this case, the advection equation becomes

$$\frac{\partial c}{\partial t} + \omega \frac{\partial c}{\partial \theta} = 0 \tag{4.4.2}$$

The solution to this equation can be found in terms of characteristics as

$$\theta - \omega t = \text{constant}$$

and (4.4.3)

$$r = \text{constant}$$

From this solution, it is obvious that the initial concentration field will simply be rotated. The concentration field consists of a cosine hill puff defined as

$$c^0(x, y) = \begin{Bmatrix} \frac{1}{2}\left(1 + cos\ \frac{\pi R}{4}\right) & \text{for } R \le 4 \\ 0 & \text{for } R \ge 4 \end{Bmatrix} \tag{4.4.4}$$

```
Old:
        IF(CLAB.NE.CSPEC(I))THEN                                  00189700
New:
        IF(CLAB.NE.CSPEC(L))THEN                                  00189700

Old:
           WRITE(IO6,522)CLAB,CSPEC(I),I                          00189900
New:
           WRITE(IO6,522)CLAB,CSPEC(L),L                          00189900
```

Figure 4.1 Bug in subroutine RDICON.

```
Old:
      CLABEXP(1:1)='V'                                                  01110500
New:
      CLABEXP='V-LEV'                                                   01110500
      WRITE(CLABEXP(6:8),'(I3)')IZ

Old:
      CLABEXP(1:5)='WFACE'                                              01110900
New:
      CLABEXP='W-LEV'                                                   01110900
      WRITE(CLABEXP(6:8),'(I3)')IZ

Old:
      CLABEXP='ZI'                                                      01115000
New:
      CLABEXP='HTMIX'                                                   01115000

Old:
      CLABEXP='EL'                                                      01115600
New:
      CLABEXP='XMONIN'                                                  01115600

Old:
      CLABEXP='TEMPK'                                                   01117800
New:
      CLABEXP='TEMPSS'                                                  01117800

Old:
      CLABEXP='RHO'                                                     01118300
New:
      CLABEXP='RHOSS'                                                   01118300

Old:
      CLABEXP='QSW'                                                     01118800
New:
      CLABEXP='QSWSS'                                                   01118800

Old:
      CLABEXP='IRH'                                                     01119300
New:
      CLABEXP='IRHSS'                                                   01119300
```

Figure 4.2 Changes in subroutine RDMET.

```
Old:
        CONAVG(I,J,LL)=CONAVG(I,J,LL)+FULCON(I,J,K,L)                    01477700
New:
        CONAVG(I,J,LL)=FULCON(I,J,K,L)                                   01477700

Old:
        CONAVG(I,J,LL)=XSUBI*CONAVG(I,J,LL)                              01480000
New:
        CONAVG(I,J,LL)=CONAVG(I,J,LL)                                    01480000
```

Figure 4.3 Changes in subroutine TIMEAV.

```
Old:
      DATA LHORIZ/.TRUE./,LVADV/.TRUE./,LVDIFF/.TRUE./,LCHEM/.TRUE./,    00022900
New:
      DATA LHORIZ/.TRUE./,LVADV/.TRUE./,LVDIFF/.TRUE./,LCHEM/.FALSE./ ,  00022900
```

Figure 4.4 The parameter change in the main program CALGRID to suppress the chemistry.

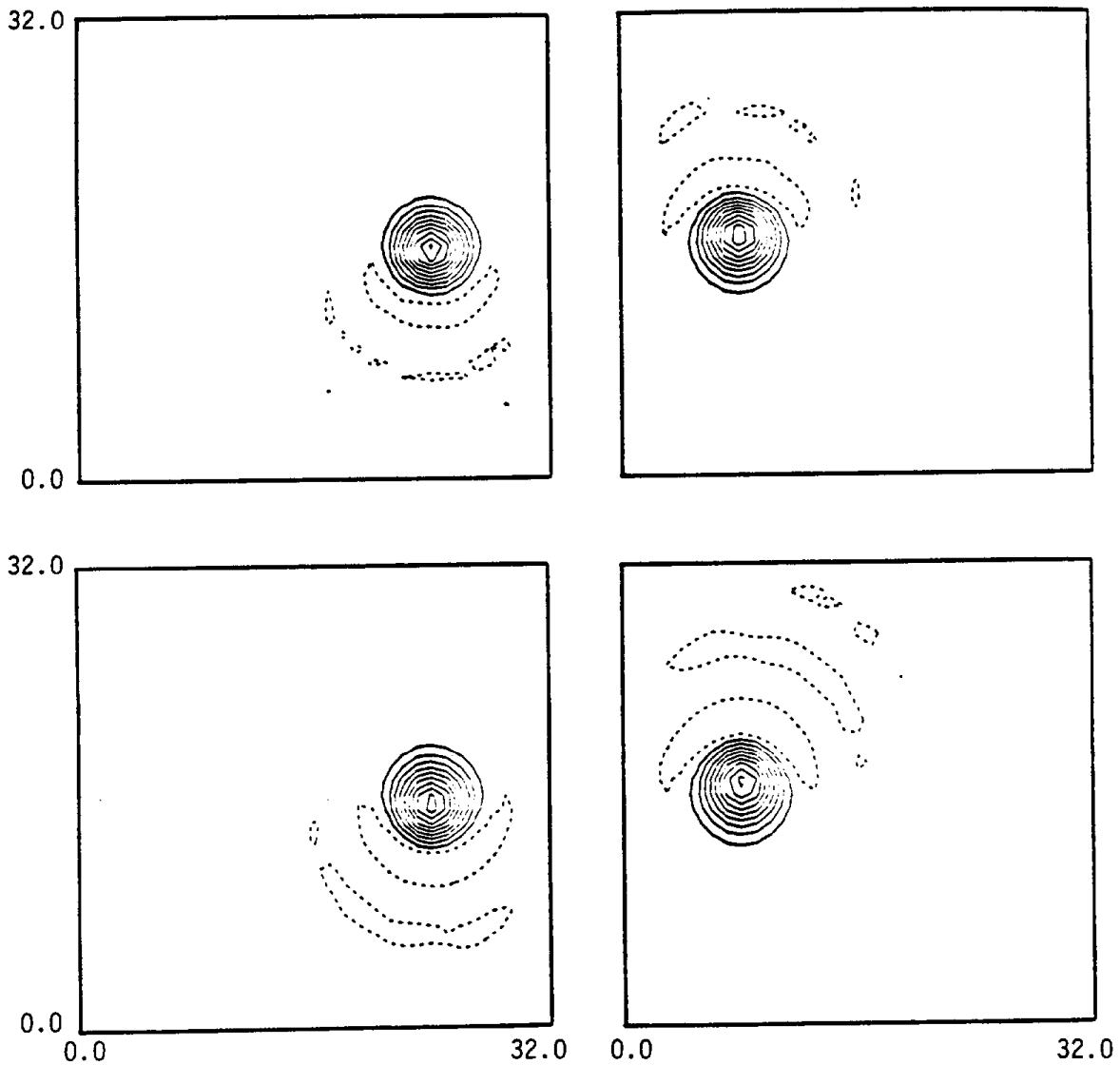


Figure 4.5 Solution obtained from CALGRID to the rotating puff problem when the filters are suppressed: after one-half (top-left), one (top-right), one and a half (bottom-left), and two (bottom-right) rotations. The peak concentrations are 102.1, 97.0, 95.4, and 93.7 respectively. The corresponding most negative concentrations are -4.1, -7.4, -11.1 and -13.5.

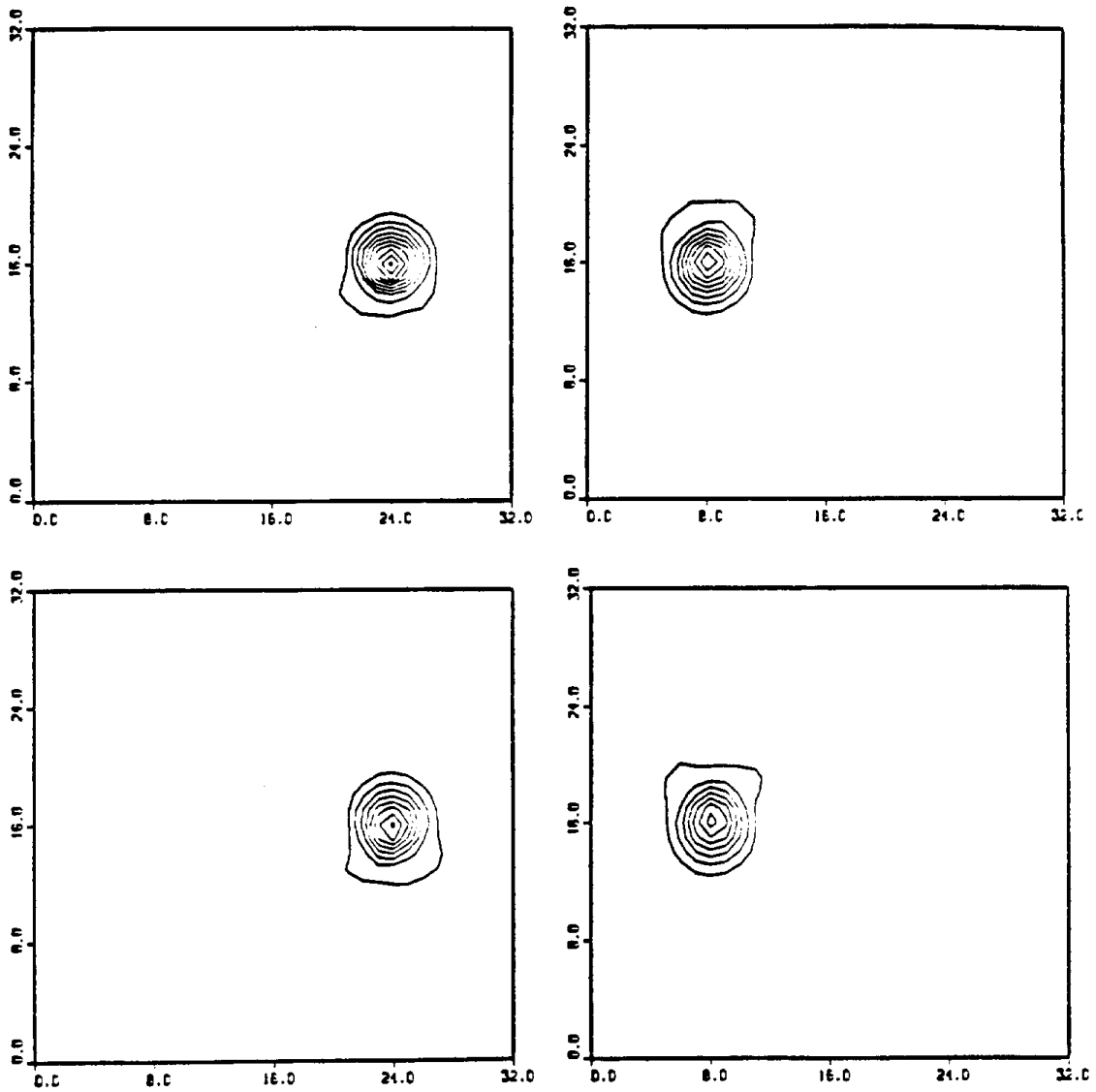


Figure 4.6 Solution obtained from CALGRID to the rotating puff problem when the filters are active: after one-half (top-left), one (top-right), one and a half (bottom-left), and two (bottom-right) rotations. The peak concentrations are 103.3, 90.5, 82.1, and 74.7 respectively.

where,

$$R = \sqrt{(x - x_0)^2 + (y - y_0)^2} \tag{4.4.5}$$

and (x_0, y_0) is the initial location of the peak. In equation (4.4.5) the base radius of the puff is equal to 4. All tests are carried until the puff completes two full rotations. The accuracy will be assessed by the peak retention capability, also defined as the maximum absolute error.

$$\text{Maximum absolute error} = \max\left(\left|c_i - c_i^e\right|\right) \tag{4.4.6}$$

where c_i^e is the exact solution value of the concentration at grid point i.

A 33x33 uniform grid with a single vertical layer is used in this test. The peak of the cosine hill shaped puff is initially located at the grid point (9, 17). A binary file, ICON.DAT, is used to input this concentration field. A bug was discovered in subroutine RDICON and was corrected as shown in Figure 4.1. The velocity field is input through the binary meteorological data file CALMET.DAT. The angular velocity, ω, is adjusted so that one complete rotation of the puff is performed in 240 time steps. This corresponds to a Courant number of $\pi/15$ at the peak of the puff. While new three-dimensional meteorological fields were generated some changes needed to be made in subroutine RDMET (Figure 4.2). Also since the hourly averaging of the concentrations created a very diffusive effect, averaging was suppressed in subroutine TIMEAV as shown in Figure 4.3. The horizontal diffusivity is set to zero to make the problem one of pure advection. The chemistry is suppressed by setting the logical variable LCHEM to .FALSE.in the main program (Figure 4.4). Since there is only one vertical layer, no vertical transport takes place.

In order to evaluate diffusion and dispersion errors associated with the Chapeau scheme, the filters are suppressed in the first test. As may be seen in Figure 4.5, the solution contains some negative concentrations due to a small amount of numerical dispersion that follows in the wake of the advected distribution (represented by dashed contours). The maximum negative concentration in the field after two complete revolutions is equal to 13.5% of the original peak height. Also of interest is the overshoot of the peak by as much as 3.1%, in the first half of the rotation. The Chapeau function scheme is mass conservative, however, being a one-dimensional method may cause overshoots at the peak.

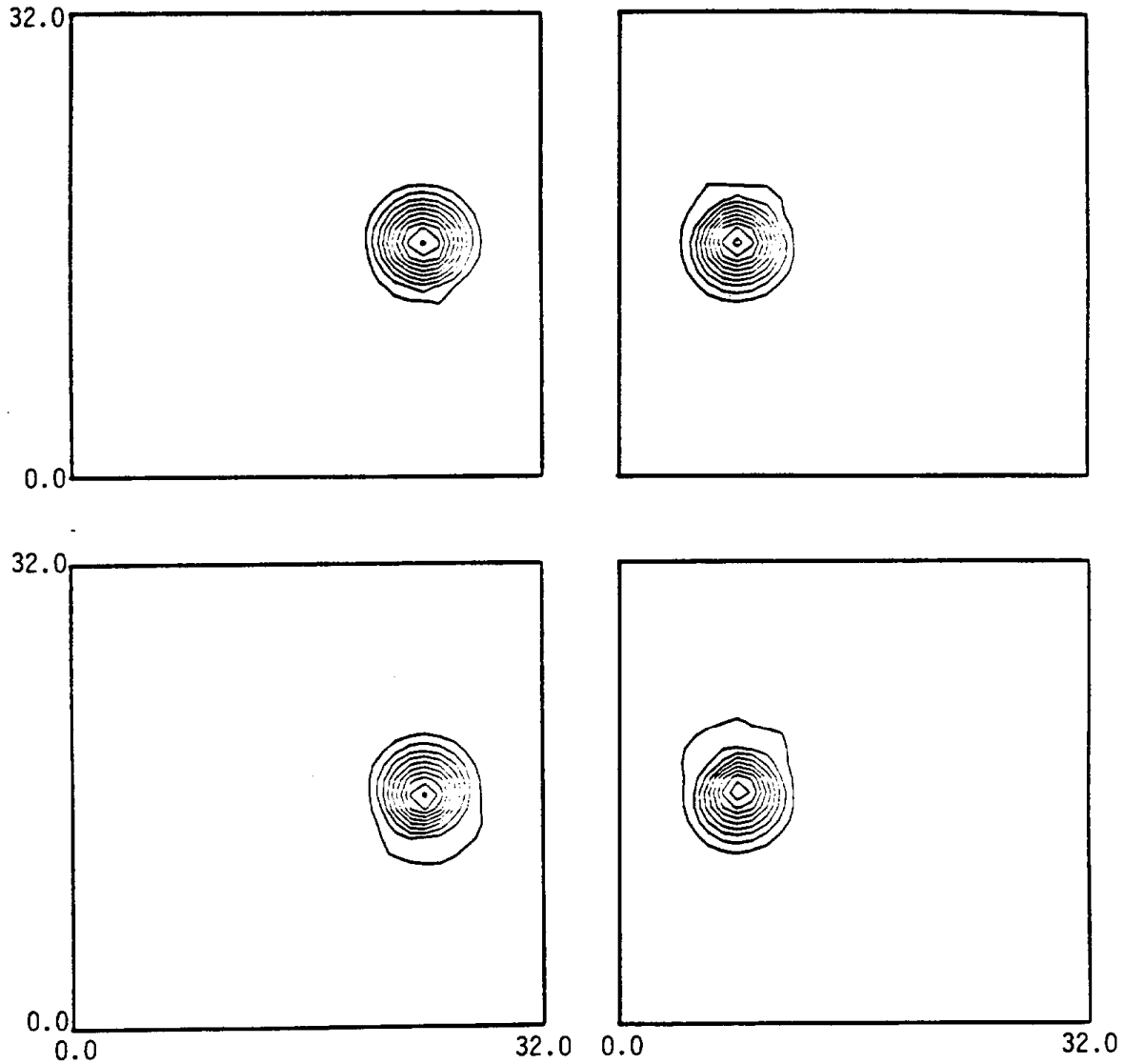


Figure 4.7 Solution obtained from CALGRID to the rotating puff problem when the puff has a base radius of 5: after one-half (top-left), one (top-right), one and a half (bottom-left), and two (bottom-right) rotations. The peak concentrations are 102.0, 103.8, 102.2, and 98.3 respectively.

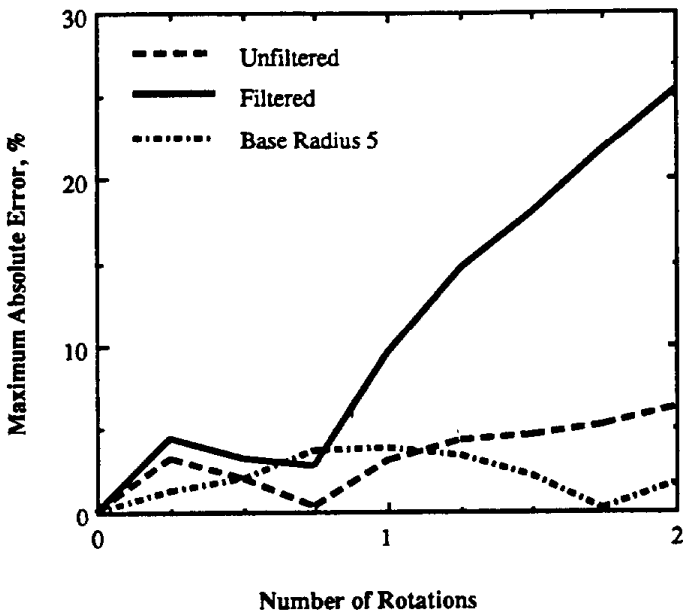


Figure 4.8 Peak retention history in rotating puff test.

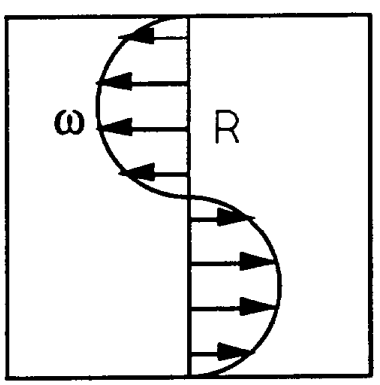


Figure 4.9 Parabolic angular velocity profile.

When the filters are activated, the solution shown in Figure 4.6 is obtained. Although the filters remove the negative concentrations in the field, the peak concentration dropped by 25.3% after two complete rotations compared to the 6.3% of the unfiltered solution. Although the Forester filter is local, and does not effect the peak heights directly, diffusion at the base (i.e., where the noise waves reside) changes the shape of the puff. The puff becomes more and more steep and the Chapeau function scheme diffuses the peak heights more and more in return. The overshoot at the peak still exist (as much as 4.3% in the first half of the rotation), because the filters are effective only in the base of the puff. The same test was carried one more time using a puff of base radius 5. The solution for this case is shown in Figure 4.7. The peak heights are plotted as a function of time both for the unfiltered and filtered cases in Figure 4.8. Also shown is the peak height history for the puff of base length 5. The latter is a less severe case with a smoother gradient yielding more accurate solutions, and the error in predicting the peak height remains within 4%.

4.5 Parabolic Tangential Velocity Profile

As mentioned above, the rigid body rotation field yields to constant velocity when the horizontal transport is split into one-dimensional operators. A more severe test problem would be one where the tangential velocity has a parabolic profile. With this field, the components of velocity along straight lines are not constant. This problem is more realistic because the wind speeds in the atmosphere are variable. The velocity field is defined as

$$u_r = 0$$
$$u_\theta = \frac{4\omega r^2}{R}\left(1 - \frac{r}{R}\right) \tag{4.5.1}$$

where R is the distance to the boundary of the domain. The tangential velocity varies parabolically as a function of r as shown in Figure 4.9.

The advection equation in this case is

$$\frac{\partial c}{\partial t} + \frac{4\omega r}{R}\left(1 - \frac{r}{R}\right)\frac{\partial c}{\partial \theta} = 0 \tag{4.5.2}$$

and the solution can be written in terms of the characteristics as

$$\theta - \frac{4\omega r}{R}\left(1 - \frac{r}{R}\right)t = \text{constant}$$

and (4.5.3)

$$r = \text{constant}$$

The initial conditions are same as before, i.e., a cosine hill with a base radius of 4 with the peak centered at grid point (9, 17) on a 33x33 domain. A new meteorological file has been created to input the velocity field. All other parameters are the same as the rotating puff test.

The exact solution to the problem is shown in Figure 4.10. The velocity field is inverted after one full rotation of the puff and the problem is continued until the puff returns to its original position. Of course, the shape of the puff must also be recovered. The solution obtained from CALGRID is shown in Figure 4.11. Especially in reverse rotation, the shape of the puff is distorted significantly. Also shown are the solutions from a two-dimensional transport operator (Odman and Russell, 1991ab) in Figure 4.12. This scheme uses the high-order accurate streamline-upwind Petrov-Galerkin (SUPG) method in conjunction with a streamline filter. The two-dimensional scheme is much more accurate than the Chapeau scheme of CALGRID. The peak retention performance of both schemes is shown as maximum absolute errors (Figure 4.13). Of note is that the SUPG scheme does not yield any overshoots. The diffusion errors grow monotonically. Even after the Chapeau function scheme overshoots the peak prediction, the peak of the puff is diminished more with the Chapeau function scheme. Only 72.6% of the peak hight remains at the end of the test while the SUPG scheme retains 81% of the original height.

As predicted, the difference in accuracy between CALGRID and a two-dimensional scheme becomes more obvious with the parabolic angular velocity profile problem. Since the velocities vary along straight lines in actual meteorological fields, the rotating puff test may not always be reliable to assess the accuracy of a transport scheme, especially if the scheme is using one-dimensional operator splitting. To our knowledge, this is the first time the parabolic angular velocity profile is used in performance evaluation of transport schemes. However, the results show that it is a better alternative to the standard rotating puff test

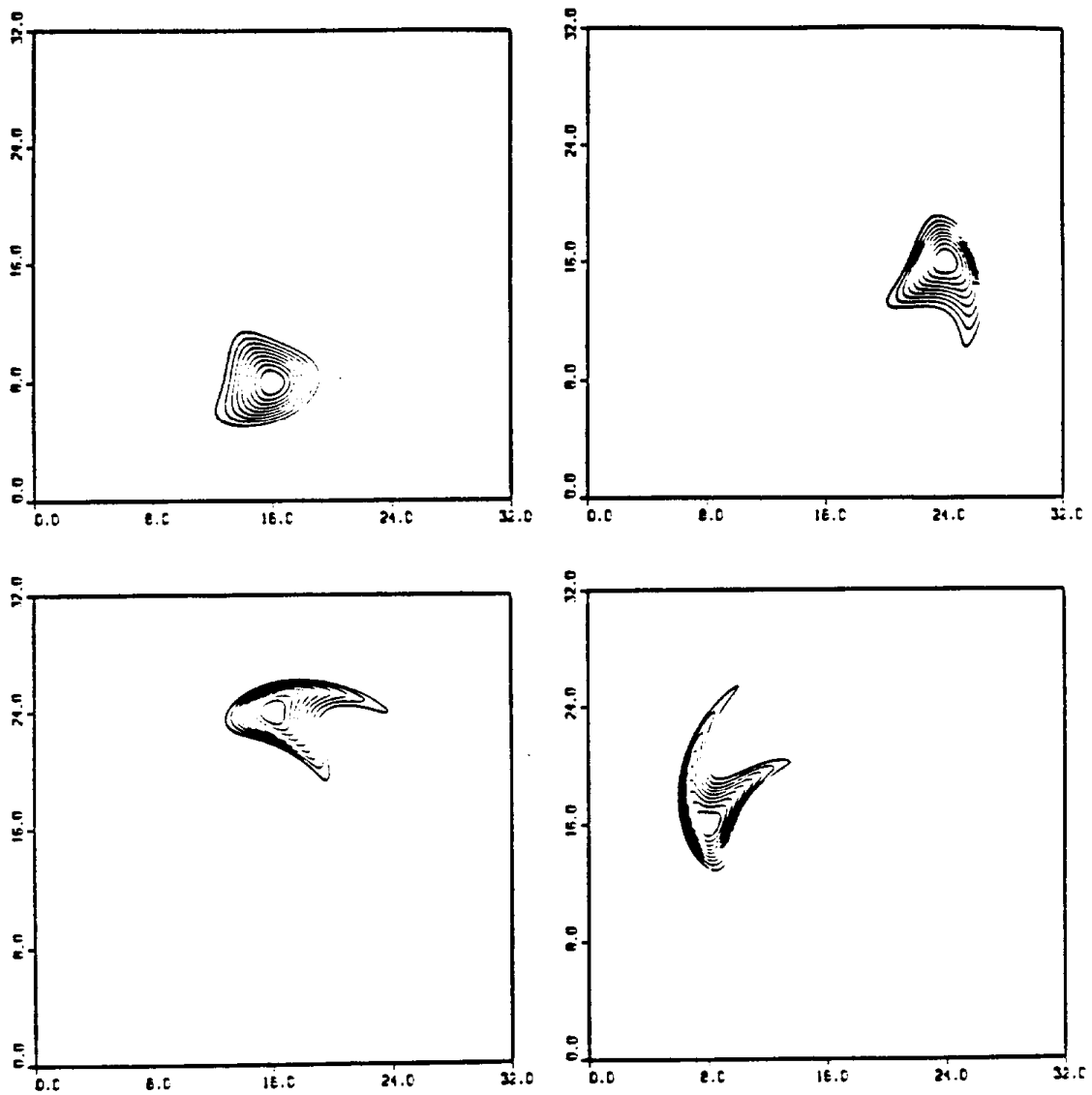


Figure 4.10 Exact solution to the parabolic angular velocity profile problem: after one-fourth (top-left), one-half (top-right), three-fourth (bottom-left), and one (bottom-right) rotations. In the second half of the problem the velocity field is inverted. The solution is the same but, this time, in reverse order.

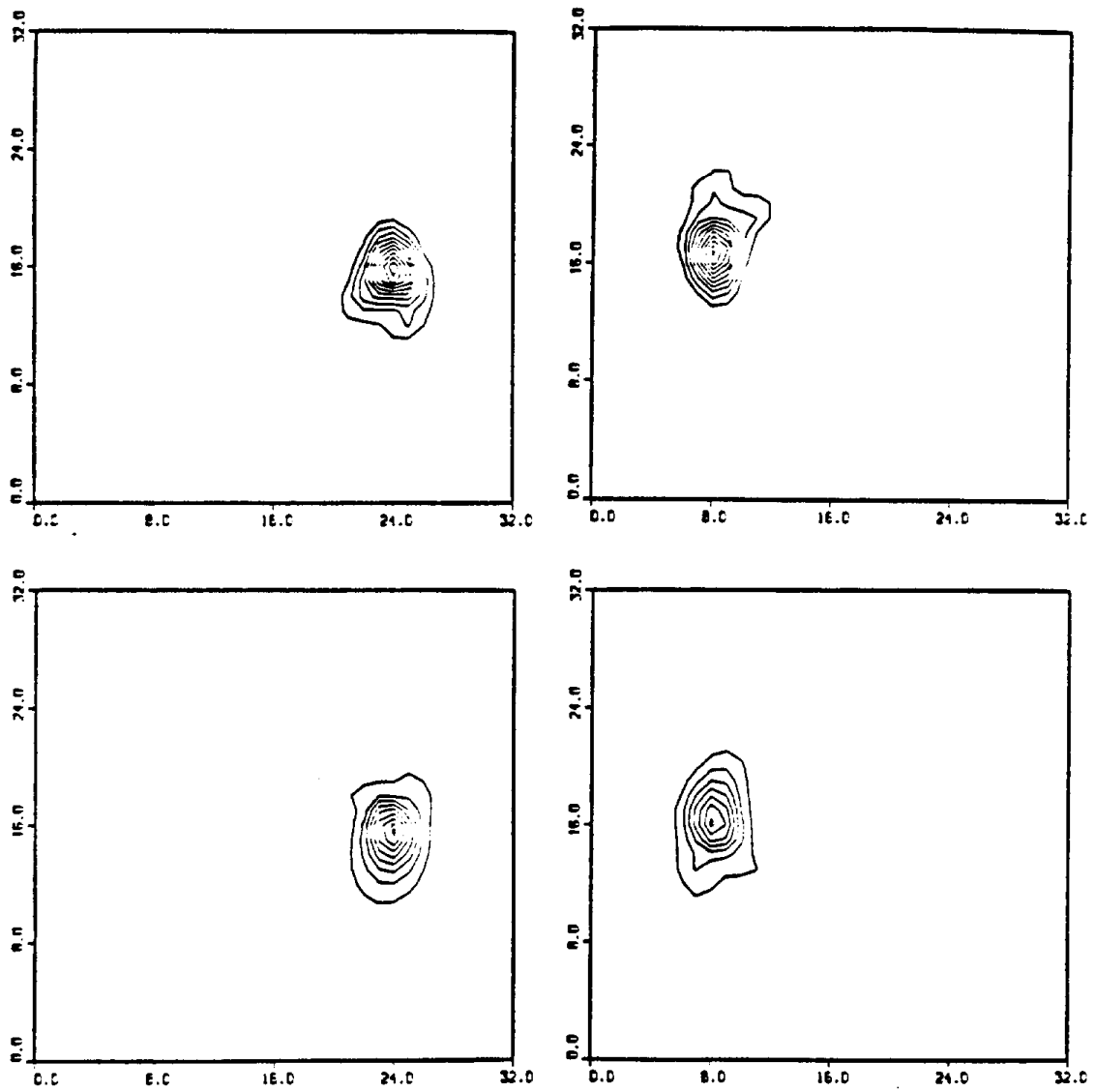


Figure 4.11 Solution obtained from CALGRID to the parabolic angular velocity profile problem: after one-half (top-left) and one (top-right) forward rotations and then , one-half (bottom-left), and one (bottom-right) backward rotations. The peak concentrations are 109.9, 104.9, 83.5, and 72.6 respectively.

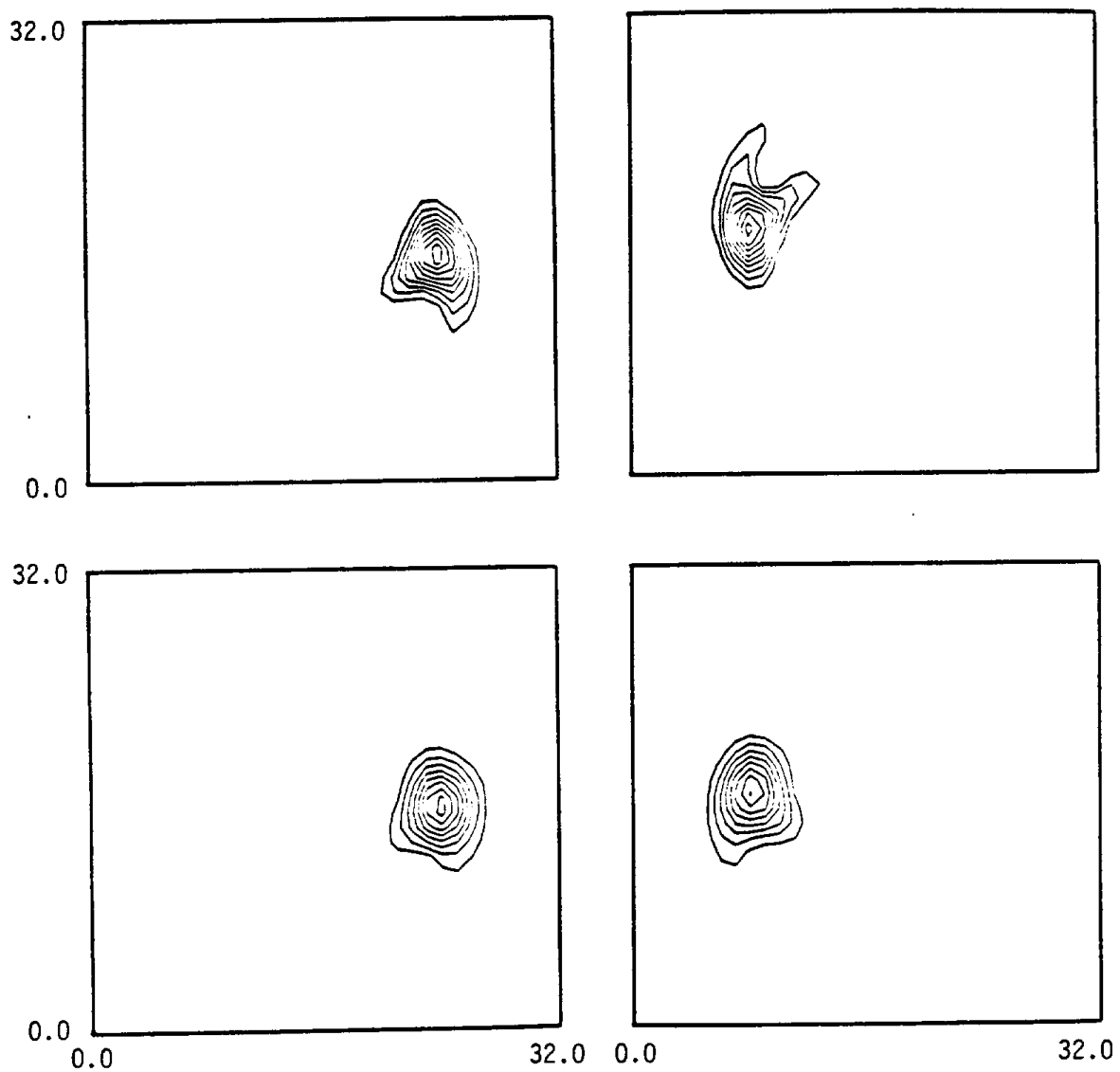


Figure 4.12 Solution obtained from two-dimensional SUPG scheme to the parabolic angular velocity profile problem: after one-half (top-left) and one (top-right) forward rotations and then , one-half (bottom-left), and one (bottom-right) backward rotations. The peak concentrations are 97.4, 95.8, 84.4, and 81.0 respectively.

4.6 Horizontal Transport and Chemistry

The performance of transport schemes are usually evaluated in advection tests. These tests investigate certain measures such as the degree of preservation of the peaks and the generation of ripples. There are some characteristics of transport schemes, however, that an advection test alone can not reveal. The nonlinear chemistry alters the shape of a pollutant puff constantly; it can change its slope, making it less or more steep, or it can even invert a peak. The ability of the scheme to adapt to such changes can only be seen and evaluated in tests with chemistry.

For test purposes, it is better to keep the chemical mechanism simple. This way, sound physical arguments can be made more readily. However, the mechanism should still be able to yield the kind of numerical difficulties encountered in photochemical models. One such simple description of the atmosphere was used by Hov et al. (1989) and is given in Table 4.1 The test itself consists of advecting puffs of different species while they react according to the chemistry described. The solution to the advection part is known: the concentration c_1 of a certain species located at some point $\mathbf{X}_1$ is moved to some other point, say $\mathbf{X}_2$, after a certain time. On the other hand, if no advection is taking place, but the chemistry is activated, the concentration at $\mathbf{X}_1$ will be changed to c_2 during the same time period. Therefore, if both advection and chemistry are applied, then the concentration at $\mathbf{X}_2$ should be c_2.

A 33x33 grid and a rigid-body rotation velocity field are used for this problem. The angular velocity ω is adjusted so that one rotation is completed in 24 hours. The horizontal transport operators is applied every 150 s and the chemistry operator, every 300 s. The solar zenith angle, θ, that appears in the photolysis reactions of Table 4.1, is held constant at 71.5° throughout. This is the angle whose cosine corresponds to the average during an equinox day in the equator. Four species, HC, HCHO, NO and NO_2, are initialized as perfect cone-shaped puffs with the peak located at grid point (9, 17) and the base radius equal to 4. The initial concentrations at the peaks are given in Table 4.2. Background concentrations are set equal to 2.5% of the peak height for these species. All other species have the same background concentrations at all nodal points.

In order to accommodate the simple chemistry, subroutine BLDUP has been replaced with the one shown in Figure 4.14. For the same purpose subroutine DIFUN is changed as shown in Figure 4.15. Also, certain parameters and the species names were changed in the main program CALGRID (Figure 4.16). The number of species is now 13

Table 4.1

Simplified chemical mechanism used in the model.

(1)	$HC + OH \rightarrow 4\,RO_2 + 2\,HCHO$	$k_1 = 6.0 \times 10^{-12}$
(2)	$HCHO + h\nu \rightarrow 2\,HO_2 + CO$	$J_2 = 7.8 \times 10^{-5}\, e^{-0.87/\cos\theta}$
(3)	$RO_2 + NO \rightarrow NO_2 + HCHO + HO_2$	$k_3 = 8.0 \times 10^{-12}$
(4)	$NO + HO_2 \rightarrow NO_2 + OH$	$k_4 = 8.3 \times 10^{-12}$
(5)	$NO_2 + h\nu \rightarrow NO + O_3$	$J_5 = 1.0 \times 10^{-2}\, e^{-0.39/\cos\theta}$
(6)	$NO + O_3 \rightarrow NO_2 + O_2$	$k_6 = 1.6 \times 10^{-14}$
(7)	$O_3 + h\nu \rightarrow O_2 + O(^1D)$	$J_7 = 1.9 \times 10^{-4}\, e^{-1.9/\cos\theta}$
(8)	$O(^1D) + H_2O \rightarrow 2\,OH$	$k_8 = 2.3 \times 10^{-10}$
(9)	$NO_2 + OH \rightarrow HNO_3$	$k_9 = 1.0 \times 10^{-11}$
(10)	$CO + OH \rightarrow CO_2 + HO_2$	$k_{10} = 2.9 \times 10^{-13}$

Table 4.2

Initial concentrations.

Pollutant	Background (ppmV)	Peak (ppmV)
CO	4.06×10^{-2}	
H_2O	2.00×10^{4}	
HC	1.02×10^{-4}	4.06×10^{-3}
HCHO	5.08×10^{-5}	2.03×10^{-3}
HO_2	4.06×10^{-8}	
NO	1.02×10^{-4}	4.06×10^{-3}
NO_2	1.02×10^{-4}	4.06×10^{-3}
O(1d)		
O_3	2.03×10^{-2}	
OH		
RO_2	4.06×10^{-8}	

Table 4.3

Predicted peak concentrations.

Pollutant	Chemistry (ppmV)	Advection + Chemistry (ppmV)	Error (%)
HCHO	1.603×10^{-2}	1.504×10^{-2}	-6.2
NO	5.70×10^{-10}	5.80×10^{-10}	+1.8
NO_2	9.570×10^{-7}	9.260×10^{-7}	-3.2
O_3	4.762×10^{-2}	4.759×10^{-2}	-0.1

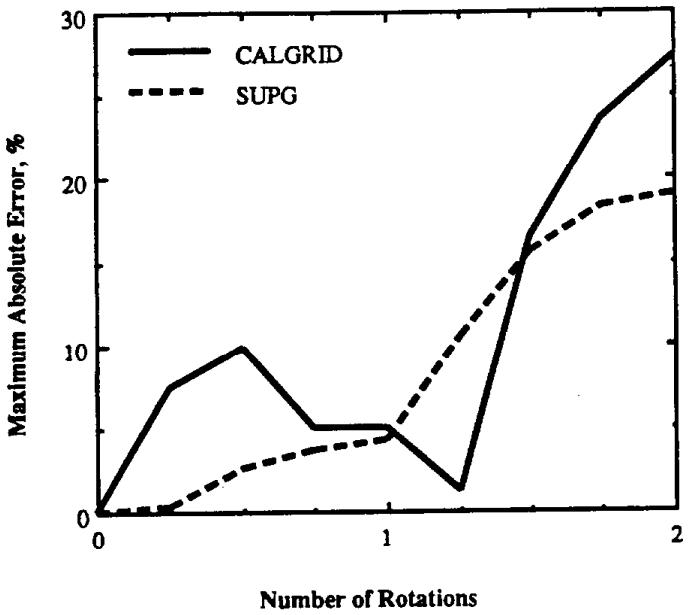


Figure 4.13 Peak retention history in parabolic angular velocity profile test.

```
C-----------------------------------------------------------------------
      SUBROUTINE BLDUP (R,CO,DCDT)
C-----------------------------------------------------------------------
C
C --- CALGRID    VERSION:  1.2     LEVEL:  890531                  BLDUP
C
C***********************************************************************
C
C     MODEL DEPENDENT SUBROUTINE TO CALCULATE FORMATION
C     RATES OF NON-REACTING SPECIES
C-----------------------------------------------------------------------
C
C --- BLDUP CALLED BY: CHMRXN, INTEGR
C --- BLDUP CALLS:     NONE
C-----------------------------------------------------------------------
C
      REAL R(*),CO(*),DCDT(*)
C
C
C        CO2
      DCDT(  1) = R( 10)
C
C        HNO3
      DCDT(  2) = R(  9)
      RETURN
      END
```

Figure 4.14 Subroutine BLDUP.

```
C-----------------------------------------------------------------------
      SUBROUTINE DIFUN (C,A,S,RK,R,CO,FR,LR)
C-----------------------------------------------------------------------
C
C --- CALGRID    VERSION:  1.2     LEVEL:  890531                  DIFUN
C
C***********************************************************************
C
C     MODEL DEPENDENT SUBROUTINE TO CALCULATE RATES OF
C     FORMATION AND LOSS FOR ACTIVE SPECIES
C-----------------------------------------------------------------------
C
C --- DIFUN CALLED BY: CHMRXN, INTEGR
C --- DIFUN CALLS:     NONE
C-----------------------------------------------------------------------
C
      REAL C(*),A(*),S(*),RK(*),R(*),CO(*),FR(*),LR(*)
C
C
C       DEFINE RATES OF REACTIONS, PARTIAL RATES
C       FOR REACTIONS WITH STEADY STATE REACTANTS
C
      R(  1) = RK(  1)*A(  8)
      R(  2) = RK(  2)*A(  7)
      R(  3) = RK(  3)*A(  6)*A(  2)
      R(  4) = RK(  4)*A(  2)*A(  4)
      R(  5) = RK(  5)*A(  3)
      R(  6) = RK(  6)*A(  2)*A(  1)
      R(  7) = RK(  7)*A(  1)
      R(  8) = RK(  8)*C(  1)
      R(  9) = RK(  9)*A(  3)
      R( 10) = RK( 10)*A(  5)
C
C      CALCULATE STEADY-STATE CONCENTRATIONS AND
C      RATES OF REACTIONS WITH SS SPECIES
C
C        O1D
C
      S(  2) = (R(  7))/(R(  8))
      R(  8) = S(  2) * R(  8)
C
C        OH
C
      S(  1) = (R(  4) + 2.*R(  8))/(R(  1)+ R(  9)+ R( 10))
      R(  1) = S(  1) * R(  1)
      R(  9) = S(  1) * R(  9)
      R( 10) = S(  1) * R( 10)
C
C       DEFINE FORMATION RATES FOR ACTIVE SPECIES
C
C        O3
      FR(  1) =  R(  5)
C
C        NO
      FR(  2) =  R(  5)
C
C        NO2
      FR(  3) =  R(  3)+ R(  4)+ R(  6)
C
C        HO2
      FR(  4) =  R(  2)*2.+ R(  3)+ R( 10)
C
C        CO
```

```
      FR(  5) =  R(  2)
C
C        RO2
      FR(  6) =  R(  1)*4.
C
C        HCHO
      FR(  7) =  R(  1)*2. + R(  3)
C
C        HC
      FR(  8) =  0.
C
C
C       DEFINE LOSS RATES OF ACTIVE SPECIES                              00722500
C                                                                        00722600
C        O3
      LR(  1) =  RK(  6)*A(  2)+ RK(  7)
C
C        NO
      LR(  2) =  RK(  3)*A(  6)+ RK(  4)*A(  4)+ RK(  6)*A(  1)
C
C        NO2
      LR(  3) =  RK(  5)+ RK(  9)*S(  1)
C
C        HO2
      LR(  4) =  RK(  4)*A(  2)
C
C        CO
      LR(  5) =  RK( 10)*S(  1)
C
C        RO2
      LR(  6) =  RK(  3)*A(  2)
C
C        HCHO
      LR(  7) =  RK(  2)
C
C        HC
      LR(  8) =  RK(  1)*S(  1)
C
C
      RETURN                                                             00733200
      END                                                                00733300
```

Figure 4.15 Subroutine DIFUN.

```
Old:
      DATA NSPEC/47/,NSA/36/,NSDD/15/,NSE/13/                           00017800
New:
      DATA NSPEC/13/,NSA/10/,NSDD/0/,NSE/1/                             00017800

Old:
      DATA CSPEC/'CO2','HO2H','RO2-HO2-PROD','RO2-RO2-PROD','-OOH',     00018100
     1 '-C','-N','H2','H2SO4','O3','NO','NO2','NO3','N2O5','HNO3',      00018200
     2 'HONO','HNO4','HO2','CO','RO2.','RCO3.','PAN','HCHO','RNO3',    00018300
     3 'MEK','CCHO','MGLY','CRES','AFG2','AAR1','AAR2','AAR3','AAR4',  00018400
     4 'ETHE','OLE1','SO2','HO.','O','O*1D2','RO2-R.','RO2-N.','R2O2', 00018500
     5 'RO2-XN.','HOCOO.','-NO2','O3OL-SB','H2O'/                      00018600
New:
      DATA CSPEC/'CO2', 'HNO3', 'O3', 'NO', 'NO2', 'HO2', 'CO', 'RO2',
     & 'HCHO', 'HC', 'OH', 'O1D', 'H2O'/
```

Figure 4.16 Changes in main program CALGRID due to new chemical mechanism.

```
Old:
      IRXP    = 32                                                      00467200
New:
      IRXP    = 0                                                       00467200

Old:
      OPEN(IRXP ,FILE='CALBE221.RXP',STATUS='OLD')                      00467500
New:
C     OPEN(IRXP ,FILE='CALBE221.RXP',STATUS='OLD')                      00467500

Old:
      CLOSE(IRXP )                                                      00472900
New:
C     CLOSE(IRXP )                                                      00472900
```

Figure 4.17 Changes in subroutine CHEMI.

with 10 active species. There are no dry deposited species. The number of emitted species had to be set to one due to a memory allocation problem in the code. However the emission rate has been set to zero, therefore, there are no emissions in the test problem.

The file CALBE221 is not used in this test problem. In this respect, changes shown in Figure 4.17 were made in subroutine CHEMI. In subroutine CHMRXN, the photolysis rates were made constant by eliminating the variation of the zenith angle. Then, the water vapor concentration was made constant by eliminating the related lines of code. Finally, the lumping technique used in conjunction with the original chemistry was eliminated. All these changes are shown in Figure 4.18.

The changes shown in Figure 4.19 are made in subroutine COMP. The model dependent initialization subroutine CONSTR is not used. Also, in subroutine INPQA, certain changes were necessary (Figure 4.20). The subroutine OPSPLT has been changed as shown in Figure 4.21 in order to accommodate time varying boundary conditions.

The solution to chemical kinetics is obtained by using the hybrid solver of CALGRID. Although this is not the exact solution of the problem, since the horizontal transport is not taking place, the solution has the errors involved with the chemistry only and is exact as far as the transport is concerned. The predicted advection with chemistry solutions are compared to the chemistry solution in Fig.'s 22, 23, 24, and 25 for NO, NO_2, HCHO and O_3 respectively. The predicted peak concentrations and the errors relative to the exact solution are also shown in 4.3.

Since the shape of different pollutant puffs are altered differently by the chemistry, the errors are not the same for all species. The largest shape changes occur with HCHO where the pollutant puff becomes more steep, and NO_2 where the puff is inverted, i.e., the concentration at the original peak becomes less than the background concentration. These species also display the largest errors: 6.2% and 3.2% for HCHO and NO_2 respectively. These errors are of the same order of magnitude as the diffusion errors observed in the rotating puff test. Therefore, it can be concluded that CALGRID's transport scheme performs well in conjunction with chemistry. There are no additional errors due to any mismatch, and the diffusion errors of the transport scheme prevail.

Old:

```
          CALL VRTPHK(COSZ,HT)                                          00491300
```

New:

```
C         CALL VRTPHK(COSZ,HT)                                          00491300
```

Eliminated:

```
C     UPDATE THE WATER CONCENTRATIONS IN PPM                            00492200
C                                                                       00492300
      CONC( 4) = WATCON(TEMP,RH,PRESS)                                  00492400

C                                                                       00493100
C     LUMPING FOR THE SULFUR-CONTAINING SPECIES (SO2,SULFATE)           00493200
C                                                                       00493300
           TOTSUL = CONA(27) + CONB( 9)                                 00493400
C                                                                       00493500
C                                                                       00493700
            CONB( 9) = TOTSUL - CONA(27)                                00493800
C                                                                       00493900

C     EMPLOY THE LUMPING TECHNIQUE FOR N-, S-, AND NO & O3.             00495500
C                                                                       00495600
           TOTSUL = CONA(27) + CONB( 9)                                 00495700
C                                                                       00495800
           RAT2  = LR( 1) * CONA( 1) - LR( 2) * CONA( 2)                00495900
           DIFF1 = FR( 1) - FR( 2) - RAT2                               00496000
           DIFF2 = (FR( 2)+FR( 3)+FR( 4)+2.*FR( 5)+FR( 6)+FR( 7)+      00496100
     &              FR( 8)+FR(13)+FR(15)) -                             00496200
     &             (LR( 2)*CONA( 2)+LR( 3)*CONA( 3)+LR( 4)*CONA( 4)+    00496300
     &           2.*LR( 5)*CONA( 5)+LR( 6)*CONA( 6)+LR( 7)*CONA( 7)+    00496400
     &              LR( 8)*CONA( 8)+LR(13)*CONA(13)+LR(15)*CONA(15))    00496500
           RHS1 = (CONA( 1) - CONA( 2)) + DIFF1 * DTC                   00496600
           RHS2 = (CONA( 2) + CONA( 3) + CONA( 4) + 2.*CONA( 5) +       00496700
     &             CONA( 6) + CONA( 7) + CONA( 8) +    CONA(13) +       00496800
     &             CONA(15)) + DIFF2 * DTC                              00496900
         IF(CONA( 2) .LT. CONA( 1)) THEN                                00497000
           INX  = 1                                                     00497100
           INY  = 2                                                     00497200
         ELSE                                                           00497300
           INX  = 2                                                     00497400
           INY  = 1                                                     00497500
           RHS1 = -RHS1                                                 00497600
         END IF                                                         00497700

C     EVALUATE THE CONCENTRATION OF NO2 AND O3 OR NO                    00498300
C                                                                       00498400
            CONA(INX) = CONA(INY) + RHS1                                00498500
            CONA( 3)  =-(CONA( 2)+CONA( 4)+2.*CONA( 5)+CONA( 6)+        00498600
     &                  CONA( 7)+CONA( 8)+   CONA(13)+CONA(15))+RHS2    00498700
         IF(CONA( 1) .LE. 0.) CONA( 1)=0.                               00498800
         IF(CONA( 2) .LT. 0. .AND. CONA( 3) .LT. 0.) THEN               00498900
            CONA( 3) = 0.                                               00499000
            CONA( 2) = 0.                                               00499100
         ELSE IF(CONA( 2) .LT. 0.) THEN                                 00499200
            CONA( 3) = CONA( 2) + CONA( 3)                              00499300
            CONA( 2) = 0.                                               00499400
         ELSE IF(CONA( 3) .LT. 0.) THEN                                 00499500
            CONA( 2) = CONA( 3) + CONA( 2)                              00499600
            CONA( 3) = 0.                                               00499700
         END IF                                                         00499800

            CONB( 9) = TOTSUL - CONA(27)                                00500700
```

Figure 4.18 Changes in subroutine CHMRXN.

```
Old:
C --- CURRENT VERSION OF CHEMISTRY REQUIRES 36 ADVECTED SPECIES AND      01396900
C --- 47 TOTAL (ADVECTED + STEADY-STATE) SPECIES                          01397000
      IF(NSPEC.NE.47)THEN                                                 01397100
New:
C --- CURRENT VERSION OF CHEMISTRY REQUIRES 10 ADVECTED SPECIES AND      01396900
C --- 13 TOTAL (ADVECTED + STEADY-STATE) SPECIES                          01397000
      IF(NSPEC.NE.13)THEN                                                 01397100

Old:
     1   'MUST BE 47 IN CURRENT VERSION OF CHEMISTRY -- NSPEC = ',NSPEC 01397700
New:
     1   'MUST BE 13 IN CURRENT VERSION OF CHEMISTRY -- NSPEC = ',NSPEC 01397700

Old:
      IF(NSA.NE.36)THEN                                                   01398000
New:
      IF(NSA.NE.10)THEN                                                   01398000

Old:
     1   'MUST BE 36 IN CURRENT VERSION OF CHEMISTRY -- NSA = ',NSA      01398400
New:
     1   'MUST BE 10 IN CURRENT VERSION OF CHEMISTRY -- NSA = ',NSA      01398400
```

Figure 4.19 Changes in subroutine INPQA.

```
Old:
      DO 500 I=2,NXM1                                                     00287400
      DO 500 J=2,NYM1                                                     00287500
New:
      DO 500 I=1,NX                                                       00287400
      DO 500 J=1,NY                                                       00287500
```

Figure 4.20 Changes in subroutine OPSPLT.

```
C                                                                         00273400
Old:
      LDB=.TRUE.                                                          00258800
New:
      LDB=.FALSE.                                                         00258800
```

Figure 4.21 Changes in subroutine COMP.

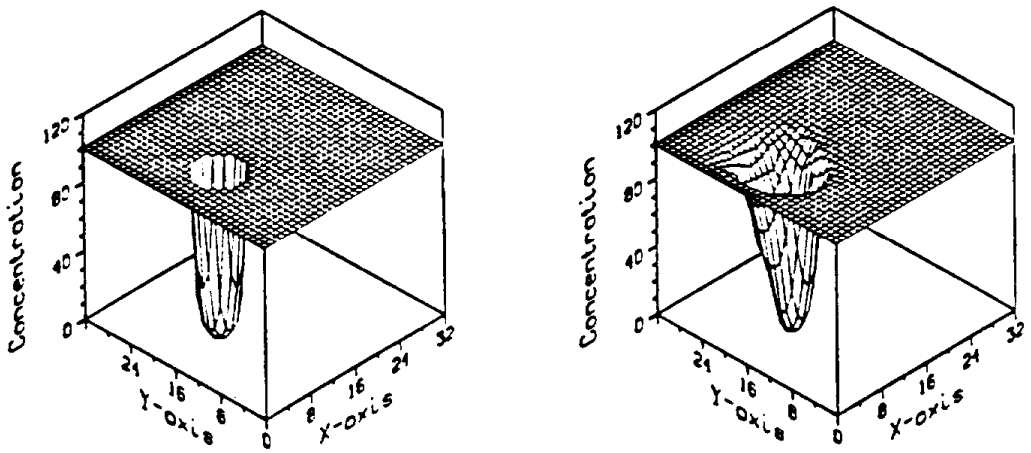


Figure 4.22 Solutions for NO of chemistry and advection + chemistry. 100% corresponds to a concentration of 5.700×10^{-10} ppmV.

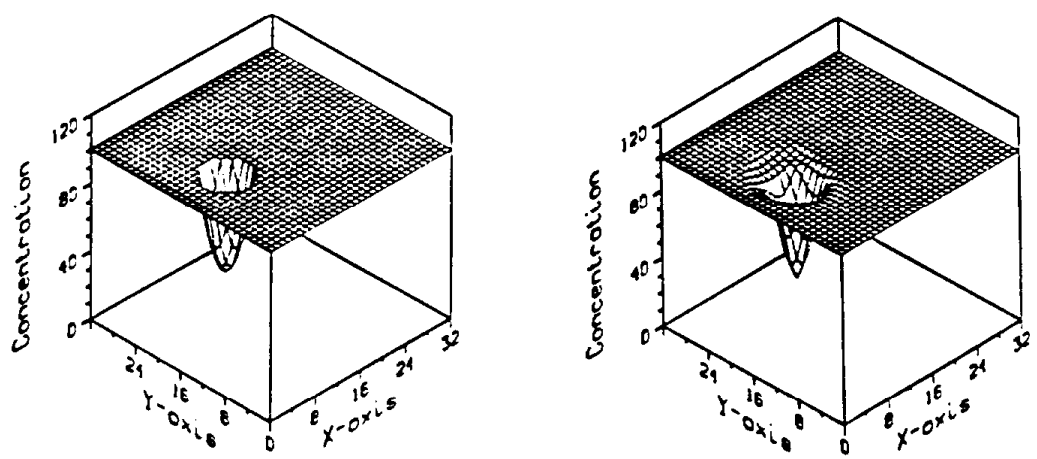


Figure 4.23 Solutions for NO_2 of chemistry and advection + chemistry. 100% corresponds to a concentration of 9.570×10^{-10} ppmV.

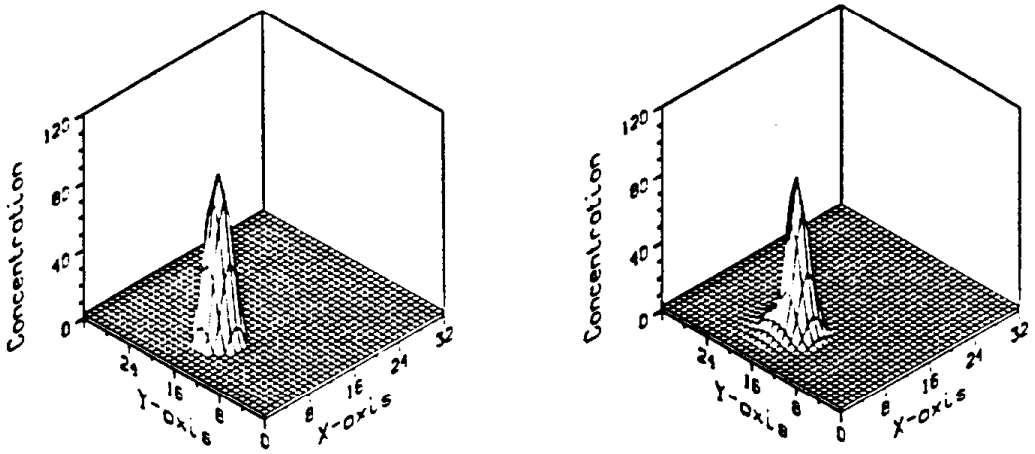


Figure 4.24 Solutions for HCHO of chemistry and advection + chemistry. 100% corresponds to a concentration of 1.603×10^{-10} ppmV.

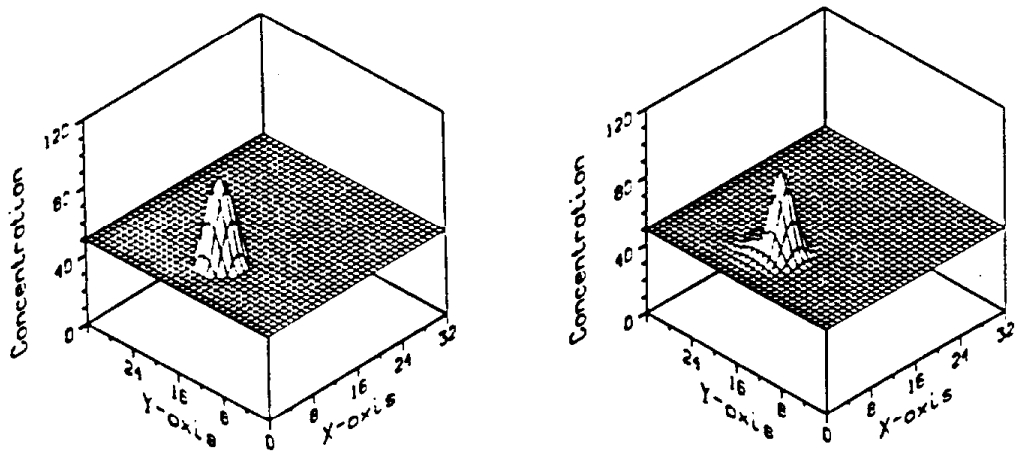


Figure 4.25 Solutions for O_3 of chemistry and advection + chemistry. 100% corresponds to a concentration of 4.762×10^{-10} ppmV.

4.7 Error Propagation in the Solution of the Chemical Kinetics

The CALGRID model has two methods for integrating the chemical kinetic formation and loss. Both are widely used, fast, numerical solvers for stiff equations, and take advantage of the solution characeristics of chemically reacting systems. The documentation tends to prefer the Quasi-Steady State Solver (QSSA; Hestvelt et al. 1978). It was found to be faster and, after the addition of a step to conserve oxidized nitrogen compouinds, more accurately tracked oxidized nitrogen species than the Hybrid solver of Young and Boris (1974). The QSSA solver is also used by the Regional Oxidant Model (Lamb, 1983), and the Regional Acid Deposition Model (Chang et al., 1988). The Hybrid solver is used by the CIT model (McRae et al., 1983; Harley et al., 1992). Both methods were tested in this study. Of note, the version of CALGRID originally supplied implemented the QSSA solver in a predictor-only mode, with specified time steps. After completion of the original tests of the fast ODE solvers, a modified version of CALGRID was supplied, with a modified QSSA solver. The modified QSSA solver used a predictor-corrector scheme, with automatic, internal choice of time step. Because the original tests had identified errors originating from having just the predictor step and fixed time steps, it was decided to extend the original tests to also assess the modified QSSA. The results of these tests are discussed below.

First, the original QSSA and hybrid solvers were evaluated for error propogation by using a series of box-model types tests. They were compared against the same calculations conducted using a very accurate, Gear type solver; LSODE (Hindmarsh, 1985). The results of those tests are discussed in detail in Odman et al. (1992), which is attached as Appendix A to this report. The treatment of nonlinear chemical kinetics requires special attention in air quality models because most of the computer time is spent in solving the equations describing the chemistry. Two fast solvers, the hybrid and QSSA schemes, were compared to the Gear method for accuracy. The conservation errors in the QSSA scheme are significant, therefore, it should be used with a mass conservative linear transformation technique. The NO_x lumping technique yields better accuracy than the proportional distribution of errors among all nitrogen containing species. The hybrid scheme, even without lumping, is about two times more accurate than the QSSA. The latter can be made more accurate by decreasing the predetermined time step, but the resulting scheme is usually less efficient than the hybrid scheme.

Simple test problems may be misleading in efficiency considerations. A moderate sized test problem with varying photolysis rates and ROG/NO_x emission ratios was designed specifically to measure the performances in vector processing mode. This problem, spanning a three day period, approximates real simulation situations better than test problems with constant rates. Both schemes displayed the largest errors around an ROG/NO_x emission ratio of 8:1, which is very close to the observed value in many urban areas. The QSSA scheme with NO_x lumping used about 4 times less computer time in solving this problem. Both schemes use about 50% of their CPU times in computing the production and loss rates. This process is not readily vectorizable and restricts the overall gain for both schemes. In vector processing mode, the integrating part of the hybrid scheme experienced approximately 50% larger speed-up than the QSSA.

These results show that there is a trade-off between the accuracy and efficiency of fast chemical kinetic solvers considered here. When the suggested time steps are used, the QSSA scheme is faster than the hybrid scheme, though less accurate. In multi-day simulations, some instability problems may be encountered. Depending on the stiffness of the problem, special techniques may be required to obtain stable solutions. These techniques may be different for different chemical mechanisms and different atmospheric conditions may require new techniques. On the other hand, the hybrid scheme gives stable solutions of better accuracy and is not restrictive computationally. Since the hybrid scheme displays better vector speed-ups, the CPU time differences between the two become smaller in vector processing mode. Thus the hybrid scheme is numerically more robust, though usually slower.

After those tests were completed, the modified QSSA solver was provided. It was decided to test that version of the solver *in situ,* by extending a separate project where CALGRID was being modified to use CB-IV, and applied to a SCAQS episode. A more detailed discussion of the these tests and the issues are contained in section 5.1 of this report. Those tests did indicate that the modified QSSA, after further modification by this group, provided acceptable accuracy and computational performance. This version of the code has been successfully implemented by Tesche (1992) in an application to the South Central Coast of California.

One issue that is not addressed in the CALGRID documentation is that the chemical kinetics and the vertical, turbulent diffusive transport have similar characteristic times. This argues for the two processes not being split into separate operators, as is done

in CALGRID. As it turns out, the characteristics of the diffusion process and the chemical decay are similar (negative exponential form), and can be solved simultaneously using the ODE solvers tested above. Simultaneous solution of the chemical kinetics and vertical, diffusion-dominated, transport should be considered in future modifications and evaluations of CALGRID.

In summary, the two ODE solvers available in CALGRID (the hybrid and modified QSSA) provide acceptable performance in a typical application. The modified QSSA and hybrid solvers have similar performance in terms of computational efficiency and accuracy (the modification significantly slowed the QSSA solver from the original tests discussed in Appendix 2). The errors would likely be on the order of 1 to 2%. As discussed below, this is below those that can originate from the transport algorithm.

4.8 Vertical Transport

The vertical transport component of CALGRID is implemented in the routine ZTRANS.F and employs a hybrid fully implicit scheme to solve the one dimensional advection diffusion equation with dry deposition and emission injection. The governing equation is

$$\frac{\partial c}{\partial t} + \frac{\partial wc}{\partial z} = \frac{\partial}{\partial z} K_{zz} \frac{\partial c}{\partial z} \tag{4.8.1}$$

where w is the vertical component of the velocity and K_{zz} is the vertical eddy diffusivity. The variation of the elements of K_{zz}, as a function of time and atmospheric stability are described in Yamartino et al. (1992). Unlike the horizontal transport case discussed in section 3, under typical daytime conditions we have

$$\text{Advective Flux (wc)} << \text{Diffusive Flux} \left(K_{zz} \frac{\partial c}{\partial z} \right) \tag{4.8.2}$$

and so the transport equation is primarily parabolic in character. The parabolic nature of the problem makes the numerical solution considerably easier. There are several types of boundary conditions for (4.8.1) and they depend upon whether the computational mesh is time-varying or fixed. For a fixed mesh the lower level boundary conditions accounting for emissions E_i and surface removal by dry a deposition velocity v^i_g are given by:

$$v^i_g c_i - K_{zz} \frac{\partial c_i}{\partial z} = E_i[x,t] \quad ; z = 0 \tag{4.8.3}$$

At the top of the airshed, well above the mixed layer Z_i, the boundary condition corresponds to the zero flux condition:

$$K_{zz}\frac{\partial c_i}{\partial z} = 0 \quad ; z = H \tag{4.8.4}$$

Closure of the system of equations requires the specification of the initial concentration profile:

$$c_i(z,0) = c_i^0 \tag{4.8.5}$$

The sytem of equations (4.8.1-5) has no analytic solution for conditions typical of those encountered in the atmosphere. For many of the simple test cases, however, analytic solutions can be found in Carslaw and Jaeger (1986) or in Nawrocki and Papa (1963). The actual numerical difference expressions used to solve the equations are presented in Yamartino et al (1992) and will not be repeated here. One unique feature of ZTRANS.f is that it can use both fixed and variable mesh spacing. For fixed mesh spacing, the particular difference technique is formally second-order accurate in space. A complete discussion of the stability of the Crank-Nicholson time stepping procedure and in particular the effect on truncation error arising from the use of a non-uniform computational mesh are described in Roache (1979).

In order to assess the likely range of errors in the predictions, several approaches were adopted. A fourth-order accurate finite element solution of the diffusion equation was developed and used as a basis for comparison against the performance of the ZTRANS.f module in CALGRID. Several simplified cases were used to test the transport module under conditions where it is possible to determine analytic solutions. Figure 4.26 presents the results of a comparison between the performance of the CALGRID module, the fourth order finite element solution (FEM) and the analytic solution after 5 hours of simulation under conditions similar to those encountered under convective mixing. The differences in the results are negligible. In fact, the concentration axis in Figure 4.26 has been expand to show any difference at all. Based on these and other tests it can be concluded that the errors arising from the vertical transport solution scheme are less than 5%.

Figure 4.27 shows the form of the error as a function of the number of grid cells. It can be seen from the plot that the results are similar to those of Chapter 2 where once the number of mesh points exceeds about 5 the incremental improvement is negligible. One point of interest is that when the log spacing is used the error is smaller. While the

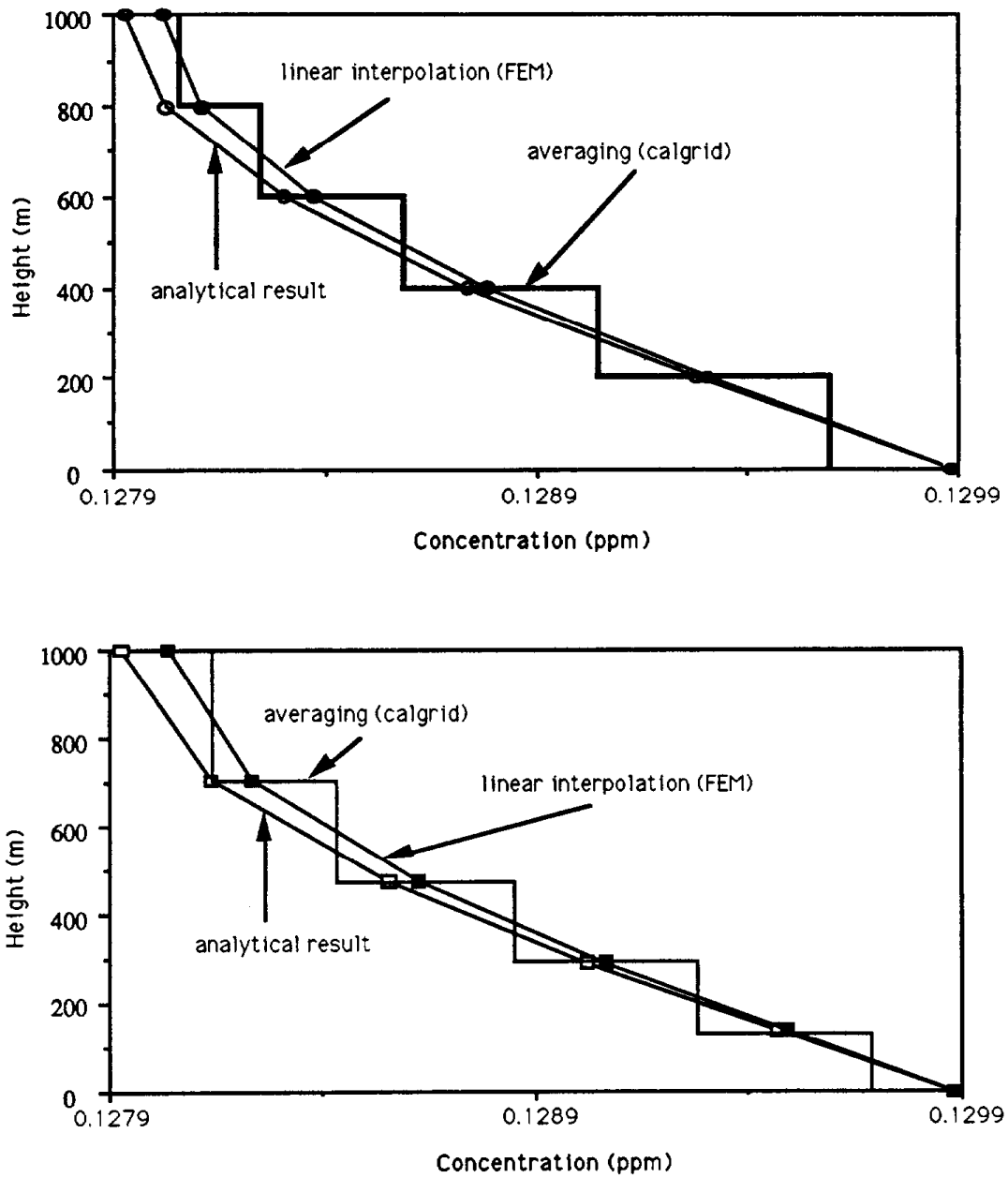


Figure 4.26 Comparison of the analytic and numerical solutions for log and linearly spaced meshes (5 vertical cells). The averaging results correspond to ZTRANS.F and the linear interpolation to a fourth-order finite element scheme (t = 5 hours).

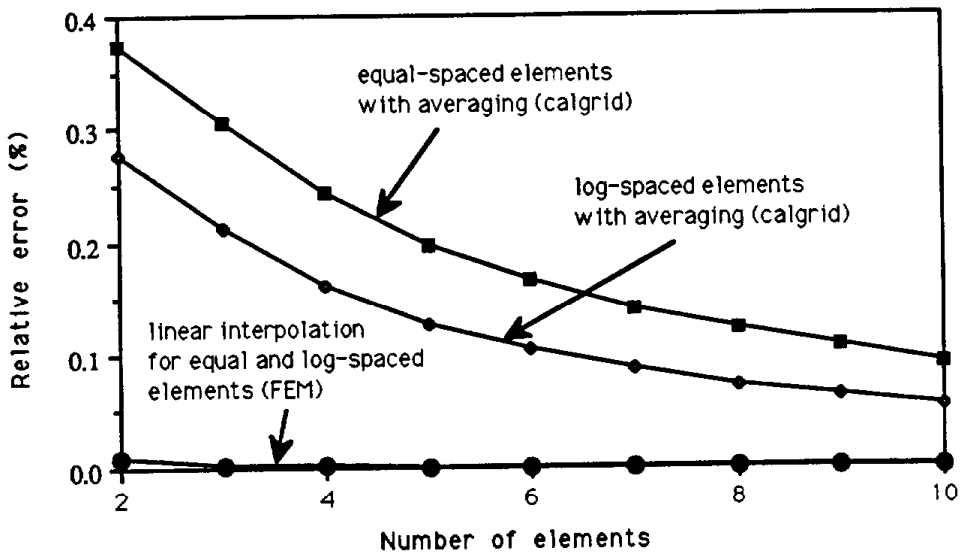


Figure 4.27 Solution error as a function of mesh spacing

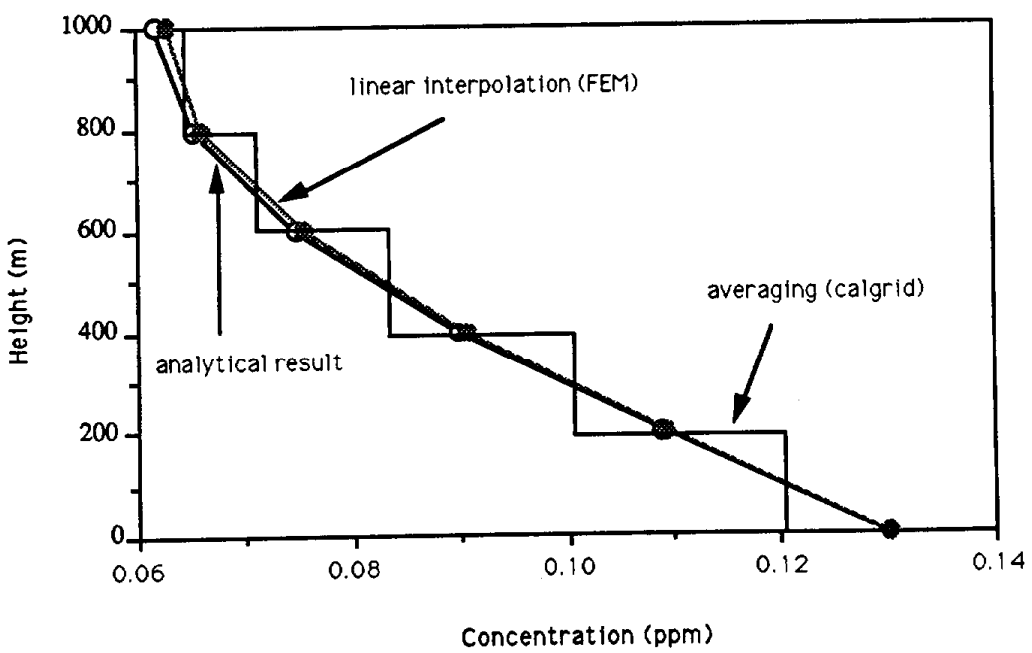


Figure 4.28 Comparison between analytic and computed vertical concentration profiles at t = 1 hour.

truncation error analysis would suggest that the variable mesh is less accurate than the fixed form the extra resolution near the ground, where there are sharp gradients in the concentration profiles, improves the overall accuracy (see Figure 4.28 for an example). In summary, under conditions typically encountered in the atmosphere the errors arising from the numerical procedures used to describe vertical transport are quite small in comparison to those observed for the horizontal transport algorithms.

4.9 Summary

The horizontal and vertical transport and chemistry components of CALGRID were extensively tested. The sources of errors stemming from the mathematical models used by these components were determined. Several test problems were used in order to identify the type and magnitude of the errors involved in each components. The components were also tested together, in order to see how errors propagate from one component to the other. The integrity of the computer code was also checked. Some inconsistencies and the changes made in order to run the test problems were reported. Simultaneous solution of the chemical kinetics and vertical, diffusion-dominated, transport should be considered in future modifications and evaluations of CALGRID.

5. Combined Error Propagation and Uncertainty Analysis

5.1 Introduction

When air quality models are used in operational practice there are many potential sources of errors in the predicted air quality levels. In the previous chapters primary attention was given to two basic types:

Structural — Errors or uncertainties in the mathematical description of the processes occurring in the atmosphere. (Chapter 2)

Algorithmic — Inaccuracies or errors that arise from the numerical solution of the governing equations. (Chapter 4)

In both of these cases problems with known analytic solutions or synthetic data were used to determine the magnitude of the errors. The utility of these tests is that they avoid the difficulty of trying to determine if the errors have arisen from the data itself. In practice however, the issue of how data errors effect the performance of the model cannot be avoided. There are two types of data related error:

Data Errors — The errors may be specific to the airshed of interest, for example in the emissions and meteorological information, or it may be in parameters incorporated into the model itself, for example in kinetic rate constants.

The data errors may also be either systematic or random. As an illustration of the difference consider the case where a reaction rate constant might have been overestimated because of an instrument malfunction. Under these conditions the higher value of the rate constant could lead to a systematic bias in the predictions of the model. A more common source of predictive uncertainty arises when there are random errors in the data. Again using the rate constant example the kinetic rate data are often reported in the scientific literature as a mean ± 1 standard deviation, with an implicit understanding that the underlying statistical distribution of the error in the rateconstant is described by a normal form. Another type of uncertaintity can arises because of spatial inhomogenities across grid cells. In a typical computational mesh the horizontal cell size is approximately 5x5 km and the associated properties such as surface roughness are assumed to be homogeneous. In actual fact there is often considerable variation in properties across the cell. An illustration of such a variation is shown in Figure 5.1.

In this chapter we introduce several different sampling procedures to assess how random errors in the data affect the predictions of the model.

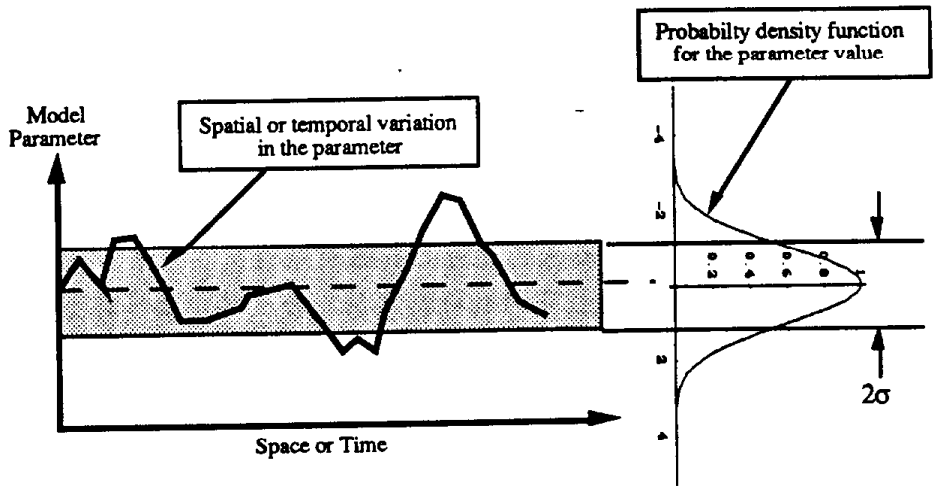


Figure 5.1 Schematic representation of how a parameter might randomly vary in space or time together with the associated probability density function. The shaded region corresponds to a $\pm$ 1 σ variation around some nominal value.

Given a knowledge of the statistical distribution of the data errors the practical issue is how to sample from the parameter or data space in such a way that we can determine the distribution of errors in the predictions. This problem is one of the most difficult in statistical analysis and is complicated by the sheer size of the parameter space that must be evaluated in airshed modeling studies.

One of the most common ways to carry out the analysis is to use Monte Carlo methods. The basic idea is to sample the parameter space and then carry out a simulation of the model for each combination. The sampling is continued until the variance in the measures used to evaluate the model's performance become stable. Some of the error metrics that might be considered are the accuracy of the peak prediction, the mean square error over the airshed, or matching the predictions at particular measurement sites. For example consider the root mean square error criteria, $<c_{rms}^i>$, at a particular location

$$<c_{rms}^i> = \sqrt{\frac{1}{N}\sum_{j=1}^{N} |c^i(\mathbf{k}_j) - c_{obs}^j|^2} \tag{5.1}$$

where $c^i(\mathbf{k}_j)$ is the model prediction of species i for parameter combination $\mathbf{k}_j$, and c_{obs} is the observation. In the Monte Carlo procedure the number of samples, N, needed to obtain stable exstimates of the error depends on the functional form of the model and the number of parameters $\mathbf{k} = \{k_1, k_2, \ldots, k_m\}$. Unfortunately the number of sample points needed to

produce stable estimates of means, let alone variances, is very large for realistic numbers of parameters. In passing it is important to note that while it is easy to define different metrics for assessing predictive errors one difficulty, that is often overlooked in practice, is the needed for information about the quality of the measurements themselves. Without some knowledge about the errors in the observational data it is often difficult to make meaningful comparisons between the predictions and measurements. There are several ways around the dimensionality problem. The Fourier Amplitude Sensitivity Test (FAST) was used by McRae et al. (1981) and Falls et al. (1979) to study the effects of errors in rate constants and stoichiometric coefficients in photochemical reaction mechanisms. Despite the efficiency of the FAST technique it is still prohibitive for models of the size of CALGRID. What is need is a better sampling procedure, one that maintains the simplicity of the Monte Carlo approach while at the same time minimizing the number of sampling points.

5.1 Monte Carlo Simulation

In order to assess the effects of random errors in the data it is necessary to introduce some metrics to characterize the effects of uncertainties. For example if $u(x,\mathbf{k})$ is some measure of the model output then its expected value, $\langle u(x,\mathbf{k})\rangle$ is given by

$$\langle u(x,\mathbf{k})\rangle = \int \cdots \int u(x,\mathbf{k})\, P(\mathbf{k})\, dk_1 \ldots dk_m \tag{5.2}$$

where $\langle . \rangle$ stands for probability distribution weighted average or the expected value and $P(\mathbf{k})$ is the probability distribution of the parameter values or data errors. The variance in $u(x,\mathbf{k})$ is given by $\sigma^2_{u(x,\mathbf{k})}$ i.e.

$$\sigma^2_{u(x,\mathbf{k})} = \int \cdots \int |u(x,\mathbf{k}) - \langle u(x,\mathbf{k})\rangle|^2\, P(\mathbf{k})\, dk_1 \ldots dk_m \tag{5.3}$$

Higher moments of the probability distribution can be derived in a similar manner. In both (5.2) and (5.3) the key need is the ability to carry out multidimensional integrals over the parameter space. There are a variety of approaches to this problem and one of the most common is to use a Monte Carlo method. Estimates of the integrals are obtained by sampling from the parameter space and then evaluating the response of the model for each parameter combination. Since the Monte Carlo method is such a well known techniques it will not be discsussed any further. For more background the reader is referred to Rubinstein (1981) and Shreider et al. (1967).

5.2 Approximation of Multi-Dimensional Integration

The key problem in assessing the effects of data errors is the evaluation of multi-dimensional integrals. In order to simplify the subsequent notation the term u(x,**k**)P(**k**) will be simplified into the equivalent form g(x) and the integrals to

$$I = \int_c \cdots \int g(\mathbf{x})\, dx_1 \cdots dx_m \tag{5.4}$$

where (5.4) is an integral over the m dimensions defining the space. The integral (5.4) is usually solved by approximating it with a discrete summation of the form

$$I = \sum w_i\, g(\mathbf{x}_i) \tag{5.5}$$

where w_i are suitable weights that depend on the sampling procedure and the functional form of g. (See Halton, 1960). There are many different schemes and perhaps the simplest is based on simply subdividing each parameter range into uniform increments. The resulting mesh of parameter values is evaluated at each intersection using a form similar to (5.5). The multidimensional mesh may be thought of as an assembly of "hyper bricks". Since the approach seems to be almost trivial and it is useful to evaluate just how effective such a procedure might be in practice.

The efficiency of such an integration formula may be gauged by considering how it fares when g(x) is the indicator-function of the hyperbrick defined by an arbitrary point A in the unit hypercube C (Halton, 1960). The practical question is of course whether a uniform placement of sample points is the best. Hammersley (1960) proposed a criterion to evaluate the relative efficiencies of different distributions of points,

$$J = \int_0^1 \cdots \int_0^1 \{S(x_1,x_2,\ldots,x_k) - Nx_1x_2\ldots x_k\}^2\, dx_1 dx_2,\ldots dx_k \tag{5.6}$$

where N is the number of points used in the estimating formula and $S(x_1,x_2,\ldots,x_k)$ is the number of these N points that fall in the hyperbrick whose upper right corner is at $(x_1,x_2,\ldots,x_k)$. If there are m samples in each direction then the number of points that must be evaluated is $N = m^k$ points, where k is dimension of the parameter space. The x values corresponding to the center of each hyperbrick are given by

$$x_i = \frac{\left(r_i + \frac{1}{2}\right)}{m} \qquad r_i = 0,1,\ldots,m-1;\ i = 1,2,\ldots,k \tag{5.7}$$

For the simple indicator function it is possible to evaluate the Hammersley criterion directly to give (Hammersley, 1960),

$$J = \left[\frac{m^2 + \frac{1}{2}}{3}\right]^k - 2\left[\frac{m^2 + \frac{1}{8}}{3}\right]^k + \left[\frac{m^2}{3}\right]^k$$
$$\approx \left(\frac{k}{3^k\, 4}\right) N^{2\frac{k-1}{k}} \quad \text{as } N \to \infty \tag{5.8}$$

For the Monte Carlo method with N points distributed at random over that unit hypercube it is a straightforward task to determine the that corresponding average value of J is given by

$$\langle J \rangle = \left(\frac{1}{2^k} - \frac{1}{3^k}\right) N \tag{5.9}$$

By comparing (5.8) and (5.9) it is possible to see that the uniform spacing approach is in fact considerably worse that the random sampling and that the Monte Carlo method is much superior for large N and k. In deriving these results no assumptions have been made about the integrand in (5.4). If information is available about the functional form of g(**x**) then it is possible to use this knowledge to develop a more efficient integration procedure. Unfortunately in the case of airshed models the functional form of g(**x**) is not known, however the key implication from the above analysis does apply. If we have to evaluate moments of the distribution in order to assess the effects of data errors on predictions then a uniform sampling procedure is not very efficient. A later section will address the question if a uniform sampling procedure is 'bad' then is it possible to find the 'best' technique.

In addition to the question related to sampling the next practical issue is how many samples are required to develop stable estimates of the moments of the model output distributions. Technically this issue is ascertaining the order of convergence of the procedure. If it is known that the integrand g(x) is quadratically integrable then the simple Monte Carlo method will have order of convergence,

$$O(N) = \frac{1}{\sqrt{N}} \tag{5.10}$$

If a set of functions, whose first-order partial derivatives,

$$\frac{\partial g}{\partial x_1}, \frac{\partial g}{\partial x_2}, \ldots, \frac{\partial g}{\partial x_k} \tag{5.11}$$

are all continuous and bounded in C, then there exists classical numerical integration or quadrature formula which have an order of convergence,

$$O(N) = \frac{1}{N^{1/d}} \tag{5.12}$$

As mentioned above, in the case of large dimension a uniform net is a poor choice. However, if the class of functions is further restricted and the requirements of continuity and boundedness of all those partial derivatives which contain no more than one differentiation with respect to each of the independent variables are fulfilled then one can construct a mesh yielding convergence of the order

$$O(N) = \frac{\ln^{k-1} N}{N} \tag{5.13}$$

Further details of the convergence properties can be found in Shreider et al. (1967).

5.3 Random Number Generator

The key conclusion from the previous section was that random sampling is a good way to evaluate multidimensional integrals. Because of this it is important to look at the procedures used for generating the random samples. The most commonly used method, and the one used for this study, is the congruential method based on recursive relationship,

$$I_{j+1} = a\, I_j + c \pmod m \tag{5.14}$$

this equation produces a sequence of integers $I_1, I_2, I_3, \ldots$, each between 0 and m-1. Here m is called the modulus, and a and c are positive integers called the multiplier and the increment respectively. The modulo notation (mod m) means that

$$I_{j+1} = a\, I_j + c - m\, k_j \tag{5.15}$$

where $k_j = [(a\, I_j + c)/m]$ denotes the largest positive integer in $(a\, I_j + c)/m$. A seed, I_0, is required for generating such pseudo-random integers. A more extensive description on the pseudo-random points generator can be found elsewhere (Rubinstein, 1981; Press et al., 1988; Halton, 1970). The generator used in this work is a function called rand(), which is available from standard ANSI C language library. The library routine rand() employs a multiplicative congruential random number generator with period 2^{32} to return successive pseudo-random numbers in the range from 0 to RAND_MAX. The default value of the symbolic constant RAND_MAX is $2^{15} - 1$.

Figure 5.2 shows the distribution of 100 sample points generated by three different techniques: uniform sampling, random sampling (Monte Carlo) and the results from a newly discoved quasi-random number generator proposed by Wozniakowsky (1991). A careful examination of the random sampling results (Figure 5.2a) shows that there are large areas or 'holes' where there are no sample points and some areas where the points are tightly clustered together.

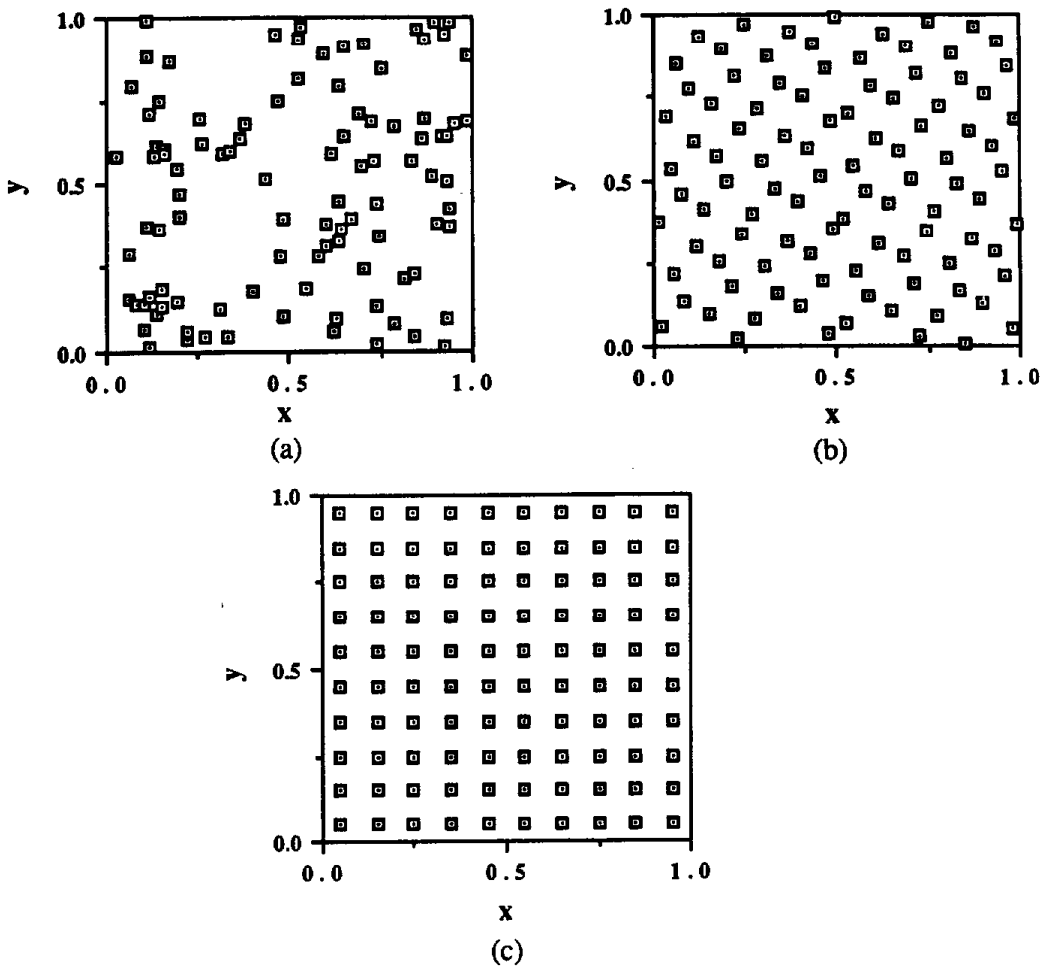


Figure 5.2 Two dimensional example of 100 (a) of pseudo-random points, (b) of Hammersley Wozniakowsky points, and (c) of uniform net.

It is beyond the scope of this report to discuss the theoretical basis for Wozniakowsky procedure but the key result is that the distribution of points shown in Figure 5.2b is in fact optimal for evaluating expected value integrals of the form (5.2). The new technique represents a major breakthough because the number of sample points needed to estimate the integral is considerably less than the classical Monte Carlo procedure. The optimal set of points resulted from a generalization of Van der Corput's sequence from two dimensions to k dimensions. A brief procedure of mechanism for generating such points will be given in the following paragraph, more detailed theoretical background is presented in Wozniakowsky (1991) and Halton (1960). The computer codes needed to use this procedure are available from the authors.

If R is any integer, then any other integer n can be written in radix-R notation as

$$n \equiv n_M n_{M-1} \ldots n_2 n_1 n_0 = n_0 + n_1 R + n_2 R^2 + \ldots + n_M R^M \tag{5.16}$$

where

$$M = [\log_R n] = \left[\frac{\ln n}{\ln R}\right] \tag{5.17}$$

the square brackets denoting the integral part. By reversing the order of the digits in n, or in other words, by taking the mirror image relative to the decimal point, one can uniquely construct a fraction lying between 0 and 1, and this function is commonly called the radical inverse function, φ, which is given as

$$\varphi = \varphi_R(n) = 0.n_0 n_1 n_2 \ldots n_M = n_0 R^{-1} + n_1 R^{-2} + \ldots + n_M R^{-M-1} \tag{5.18}$$

The Van der Corput's sequence of points in the unit square is given by $(n/N, \varphi_2(n))$ for n = 1,2,...,N. Hammersley (1960) suggested the k-dimensional sequence,

$$z_k(n) = \left(\frac{n}{N}, \varphi_{R_1}(n), \varphi_{R_2}(n), \ldots, \varphi_{R_{k-1}}(n)\right) \quad \text{for } n = 1,2,\ldots,N \tag{5.19}$$

in which he takes $R_1, R_2, \ldots, R_{k-1}$ to be the first k-1 primes.

In practical application, this sequence is not too convenient to use because the first element in the sequence (5.19) unfortunately depends on N. One way around the problem is to interative evaluate the integrals by successively adding sample points until the results converge. The new Hammersley -Wozniakowsky procedure finesses the problem by using the sample points

$$x_k(n) = 1_k - z_k(n) \tag{5.20}$$

The derivation of the computational complexity of (5.20) will not be repreated here but it suffices to say that the number of sample points needed to achieve the same level of accuracy as the Monte Carlo method is many orders of magnitude less.

5.4 Transformation of Probability Distribution

In the previous section a new procedure for generating sample points was discussed. In order to be able to evaluate the technique it is important to compare it to the conventional simulation methods. One of the difficulties encountered in practice is that routines like rand() only provide uniformly distributed random numbers. In operational practice the data values often have normal or log normal error distributions. A typical

example might be a multinormal distributed random vector where $\mathbf{x} = (x_1,\ldots,x_n)$, has a probability density function (pdf) in the form,

$$f_x(x) = \frac{1}{(2\pi)^{n/2}\,|\Sigma|^{1/2}} \exp\left[-\frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu)\right] \tag{5.21}$$

where $\mu = (\mu_1,\ldots,\mu_n)$ is the mean vector, Σ is the covariance matrix,

$$\Sigma = \begin{bmatrix} \sigma_{11} & \cdots & \sigma_{1n} \\ \vdots & \ddots & \vdots \\ \sigma_{n1} & \cdots & \sigma_{nn} \end{bmatrix} \tag{5.22}$$

The covariance matrix is positive definite and symmetric, $|\Sigma|$ is the determinant of Σ, and Σ^{-1} its inverse matrix. Equation (5.21) is commonly denoted by $N(\mu,\Sigma)$. If the errors are of the form (5.21) then there are two problems. One is how to develop normally distributed random numbers and the other is how to deal with the fact that parameters are often highly correlated.

The procedure used in this work to develop normally distributed random numbers is the Box-Muller technique that uses two independent uniform random variates from zero to one, U_1 and U_2, to obtain two independent standard normal deviates, Z_1 and Z_2, by employing the relationship,

$$\begin{aligned} Z_1 &= (-2 \ln U_1)^{1/2} \cos 2\pi U_2 \\ Z_1 &= (-2 \ln U_1)^{1/2} \sin 2\pi U_2 \end{aligned} \tag{5.23}$$

The algorithmcan be formulated in two steps as follows,

1. Generate two independent random variates $U_1(0,1)$ and $U_2(0,1)$
2. Compute Z_1 and Z_2 simultaneously by substituting U_1 and U_2 in the system of equations 5.21.

The result from this procedure are two independent standard normal deviates. An independent standard normal deviates has an expected mean value equals zero and a standard deviation equal to one. The algorithm can be obviously extended to obtain independent multinormal distributed random vector. More details can be found in Devroye (1986) and Rubinstein (1981).

The second problem is how to deal with the correlation structure. For each parameter usually μ and Σ are known but wha is required is a set of values of **x**.Since Σ is positive definite and symmetric, there exists a unique lower triangular matrix, C,

$$C = \begin{bmatrix} c_{11} & 0 & 0 \\ \vdots & \ddots & \vdots \\ c_{n1} & \cdots & c_{nn} \end{bmatrix} \tag{5.24}$$

such that

$$\Sigma = C\,C^T \tag{5.25}$$

and so the vector **x** can be represented as a transformation from uncorrelated standard multinormal random vector **z**

$$\mathbf{x} = C\,\mathbf{z} + \mu \tag{5.26}$$

The problem becomes how to to calculate matrix C if one knows the covariance matrix Σ. In this work the technique used is to employ Crout factorization. This method provides a set of recursive formulas for computation of the elements of C. The algorithm is:

1. Generate $\mathbf{z} = (z_1,\ldots,z_n)$
2. Calculate

$$c_{ij} = \frac{\sigma_{ij} - \sum_{k=1}^{j-1} c_{ik}\,c_{jk}}{\left(\sigma_{jj} - \sum_{k=1}^{j-1} c_{jk}^2\right)^{1/2}} \tag{5.27}$$

where

$$\sum_{k=1}^{0} c_{ik}\,c_{jk} = 0, \qquad 1 \le j \le i \le n \tag{5.28}$$

3. Calculate $\mathbf{x} = C\,\mathbf{z} + \mu$.

A more detailed step by step description of that method is reported by Franklin (1965) and a routine for calculating matrix C is available (Press et al., 1988).

5.5 Implementation of the Sampling Procedures

Figure 5.3 provides an overview of how individual modules in the CALGRID code can be interfaced with the two sampling procedures, the conventional Monte Carlo approach and the new Hammersley-Wozniakowsky (HW) technique. One key feature of

the approach adopted in this study, that is not obvious from the flow diagram, is the data storage technique. There are two approaches that can be employed. First, generate all the required random data points and then solve the model for each parameter combination at each time point. Given all of the time histories corresponding to each parameter combination then it is possible to determine the statistical properties of the concentration predictions at each point in time. The data storage requirements for such an approach is prohibitive. In this study we advance all the solution for each parameter combinations at the same time, compute the new concentrations and then only store the desired statistical moments. In the CALGRID model the use of a single master data array complicates the task of interfacing the system with the sampling procedures. In this study we extracted component modules and wrote interfaces for the sampling system.

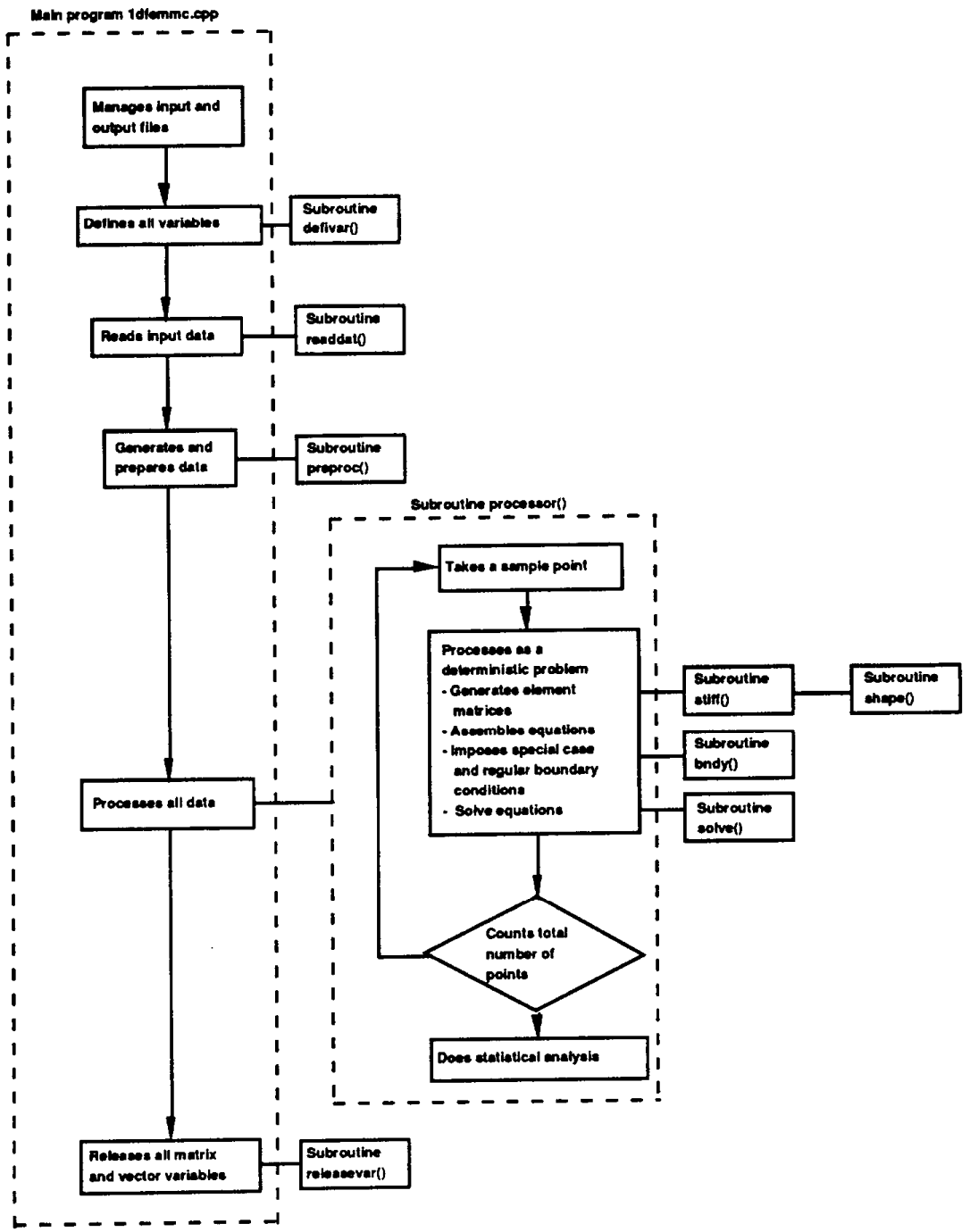


Figure 5.3 Flowchart of how the Monte Carlo and Hammersley Wozniakowsky sampling procedures can be interfaced with a CALGRID module.

5.6 Application of Sampling to Airshed Modeling Problems

In order to illustrate the sampling procedures we applied them to a modified version of the vertical transport section of the CALGRID model. The vertical transport module ZTRANS.f solves the one-dimensional diffusion equation. The code was extended to include first order chemistry for a typical species like SO_2. We chose to include this species because it has a decay time comparable to some of the longer lived organics and with a suitable transformation it is possible to develop an analytic solution to the model to test its accuracy. In the problem it is assumed that the errors in the input data are the independent normally distributed random diffusivity coefficient $D(\omega)$, the reaction constant $k(\omega)$, and the dry deposition velocity $v_g(\omega)$. The problem has mathematical statement,

$$\frac{\partial c(z,t,\omega)}{\partial t} = D(\omega)\frac{\partial^2 c(z,t,\omega)}{\partial z^2} - k(\omega)\, c(z,t,\omega) \tag{5.29}$$

with initial conditions

$$c(z,0) = c^0(\omega); \quad t = 0 \tag{5.30}$$

and boundary conditions

$$\begin{aligned} D(\omega)\frac{\partial c(H,t,\omega)}{\partial z} &= 0; \quad z = H \\ v_g(\omega)\, c(0,t,\omega) - D(\omega)\frac{\partial c(0,t,\omega)}{\partial z} &= E(t); \quad z = 0 \end{aligned} \tag{5.31}$$

The presence of variable ω in any term denotes the fact that the parameter values are random variables or a sample points from the probability distribution. The emission flux $E(t)$ in (5.31) enters the airshed model at the ground. The diurnal variation of the flux and its magnitude have been chosen to be representative of conditions occuring during the SCAQS episode. The functional form of $E(t)$ is given by

$$E(t) = a_0 + a_1 \operatorname{Sin}(\alpha_1 t + \tau) + a2 \operatorname{Sin}(\alpha_2 t) \tag{5.32}$$

and its temporal evolution is shown in Figure 5.4. The value of each of the parameters and their associated error distribution are given in Table 5.1. The numeric values are representative of daytime conditions over urban airshed and the data errors are typical of what might be encountered in practice.

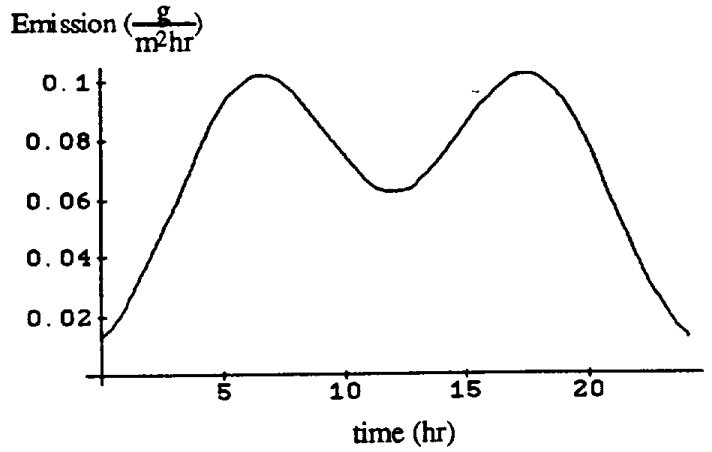


Figure 5.4 Emission profile for Vertical Transport module.

Table 5.1 Parameter Values used for Testing the CALGRID Model

Parameters	Value
Number of nodes/element	2
Number of elements	13
H	1000.0 m
c^0	0.13 ppm
$\sigma(c^0)$	0.013 ppm
D	100.0 m^2/s
$\sigma(D)$	20.0 m^2/s
k	0.14 /hr
$\sigma(k)$	0.028 /hr
v_g	1.0 cm/s
$\sigma(v_g)$	0.2 cm/s
a_0	0.039 g/m^2-hr
a_1	0.0266 g/m^2-hr
a_2	0.05 g/m^2-hr
α_1	$\Pi/6$
α_2	$\Pi/24$
τ	$-\Pi/2$

As a first step both the Hammersley-Wozniakowsky and the standard Monte Carlo methods were used to determine the effects of errors in the vertical diffusion coefficient on the diurnal concentration profile. (1000 samples points were used for both methods.) The nominal value case corresponds to setting the diffusion coefficient to its mean value and holding it fixed at that value for the whole of the diurnal cycle. This is in fact similar to a deterministic solution. The concentration mean value plot is the average value of all the 1000 concentration profiles derived from sampling different values of the diffusion coefficient probability distribution. Figure 5.5 also shows the $\pm$ 1 standard deviation envelope corresponding to all the samples.

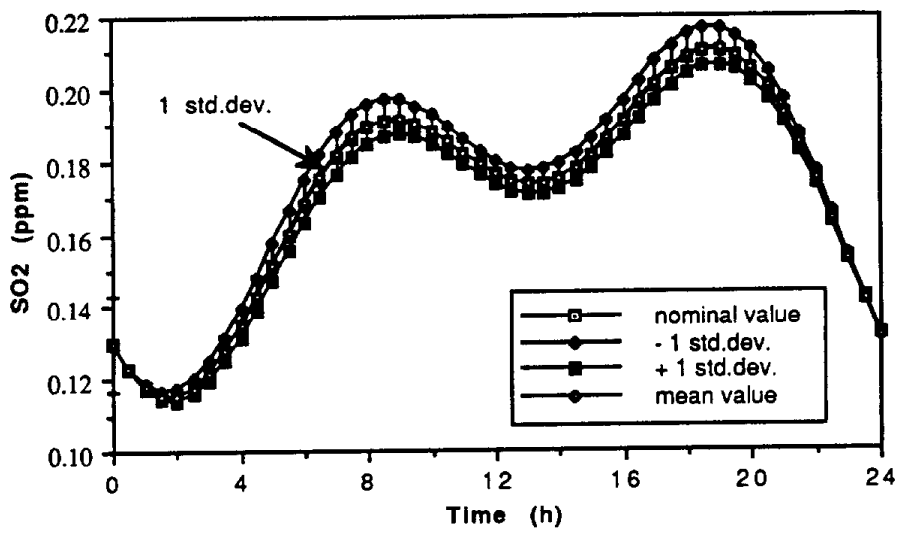


Figure 5.5 Diurnal variation of the ground level concentration of SO2 determined by both the Monte Carlo and Hammersly-Wozniakowsky sampling procedures for the case when there are only errors in the diffusion coefficient i.e. $\sigma(D) = 20$ %, $\sigma(k) = 0$ %, $\sigma(vg) = 0$ %.

There are two significant features apparent in the plot. One is that it is not possible to distinguish between the results developed using the standard Monte Carlo procedure and the Hammersley-Wozniakowsky (HW) technique. For the same level of accuracy the HW method require an order of magnitude less sample points.

Without a doubt the most striking feature of Figure 5.5, and indeed for many of the plots to follow, is that the uncertainty in the concentration predictions does not grow with time, in fact it decays later in the day. This result is of critical important to the whole issue of how to interpret and bound error propagation in airshed models. A common belief, and in fact

often confirmed for systems of ordinary differential equations, is that initial errors often undergo exponential growth to the point where the model loses any useful predictive power. In this case there are lots of constraints on the system that arise from the physical interplay between the source emissions, surface removal and chemical conversion processes. For example at the end of the day even if there are large uncertainties in the diffusion coefficient there is just not that much material left to be transported in the air column.

Figures 5.6 and 5.7 illustrate an additional important point. In both cases the effects of a systematic bias in the data was compared against the results when the parameter was treated as a random variable. Figure 5.6 presents the predicted concentration profiles for errors in the reaction rate constant. The $\pm$ one standard deviation cases correspond to fixing the rate constant at an upper ($k+1\sigma$) and a lower ($k-1\sigma$) value for the entire diurnal cycle. Under these conditions there is a approximately a maximum of 20% uncertainty in the predictions. Again it can be observed that the error grows and then decay. The systematic errors are much larger than when the reaction rate is treated as a random variable. Similar results are also observed for the dry deposition velocity case. The key finding is that if parameter values are systematically biased high or low the error in the predictions can be higher than if the parameters are treated as stochastic variables.

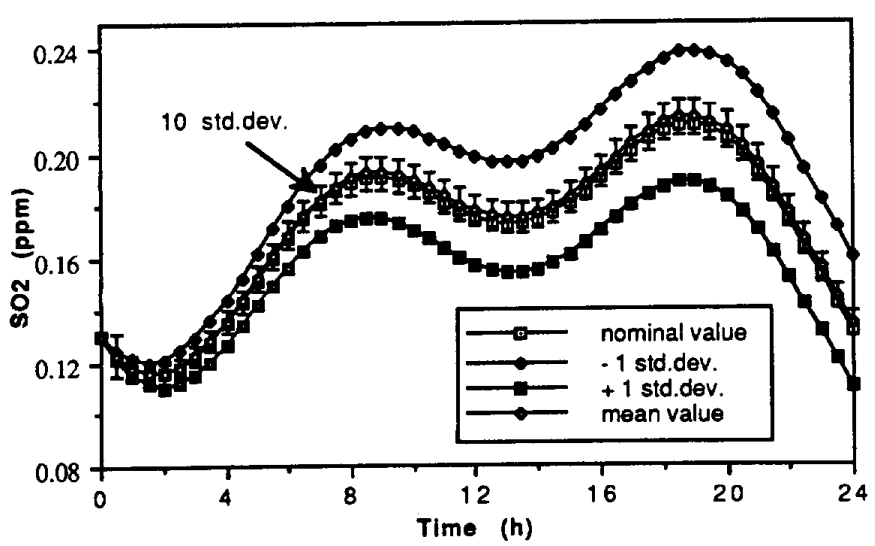


Figure 5.6 Diurnal variation of the ground level concentration of SO2 determined by sampling procedures for the case when there are only errors in the rate constant coefficient $\sigma(D) = 0$ %, $\sigma(k) = 20$ %, $\sigma(vg) = 0$ %.

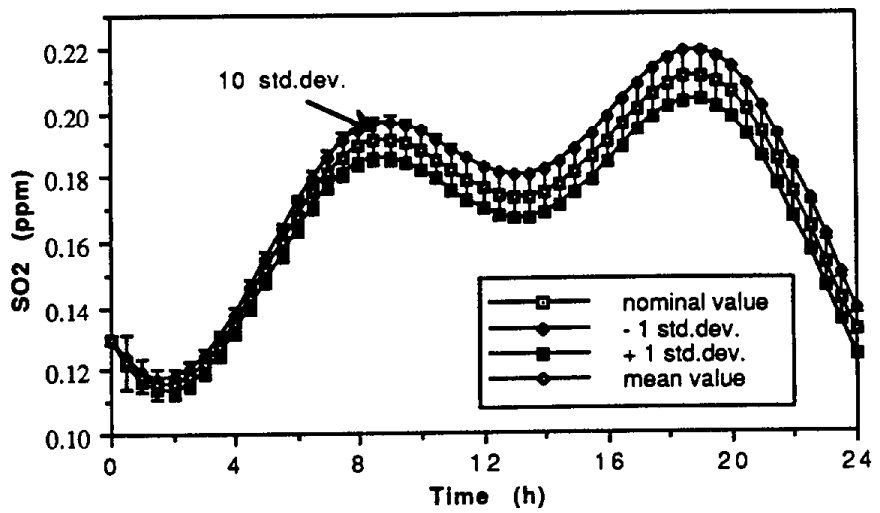


Figure 5.7 On the ground concentration of SO_2 by Monte Carlo for $\sigma(D) = 0$ %, $\sigma(k) = 0$ %, $\sigma(v_g) = 20$ %.

Figures 5.8 - 5.10 show the vertical concentration at the end of 19 hours of simulation.

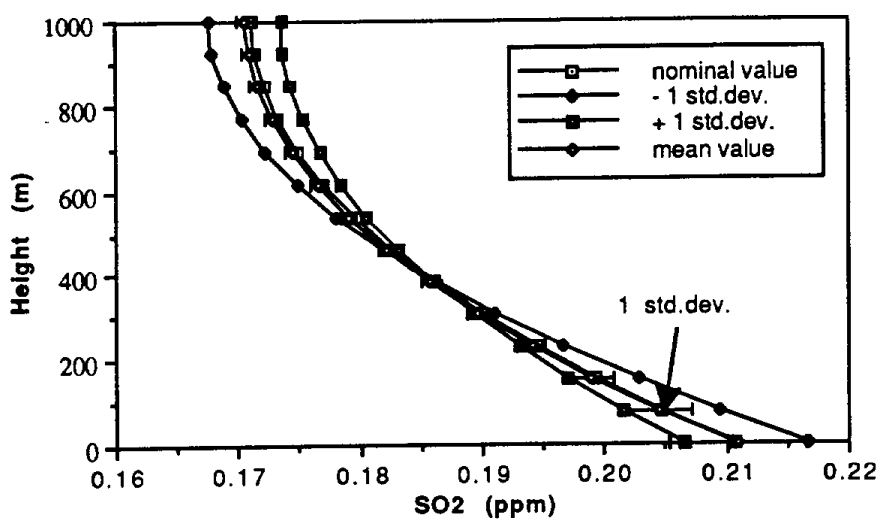


Figure 5.8 Concentration profile of SO_2 at time = 19 by Monte Carlo for $\sigma(D) = 20$ %, $\sigma(k) = 0$ %, $\sigma(v_g) = 0$ %.

The results shown in Figure 5.8 are quite consistent with the physical processes occurring in the atmosphere. If the diffusion coefficient is systematically high then the ground level concentrations are reduced and the upper level ones are increased. In the case of Figure 5.9 it can be seen if the rate constant is systematically low then the amount of material left in the vertical column is much higher. One interesting feature of Figure 5.9 is that it illustrate that the average value of the concentration, derived from the random sampling, is not necessarily the same as the profile derived using the mean value of the rate constant distribution. There are important implications of this result for interpreting the outputs when the parameters are set at the upper, nominal and lower limits. (The concentration scales used in Figures 5.9 and 5.10 have been magnified to show the variation. The actual variation in concentrations is much less than 30%.)

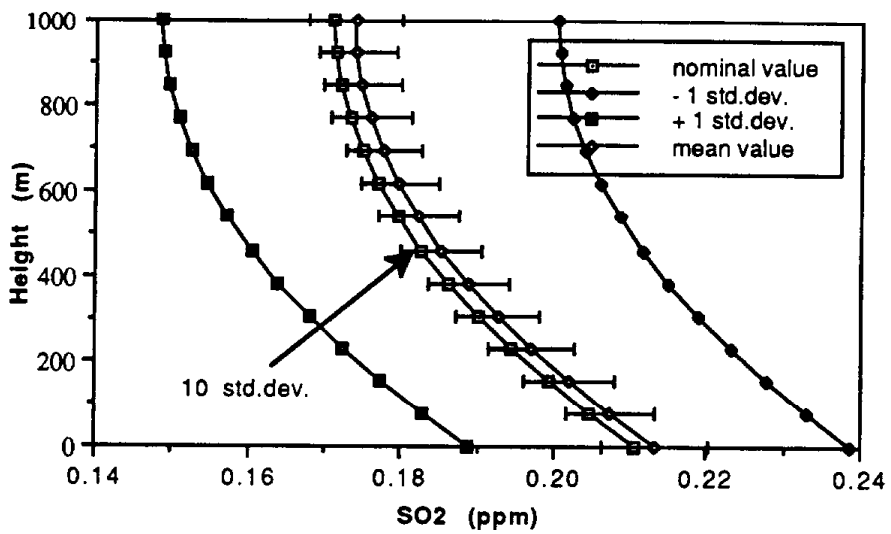


Figure 5.9 Concentration profile of SO_2 at time = 19 by Monte Carlo for $\sigma(D) = 0$ %, $\sigma(k) = 20$ %, $\sigma(v_g) = 0$ %.

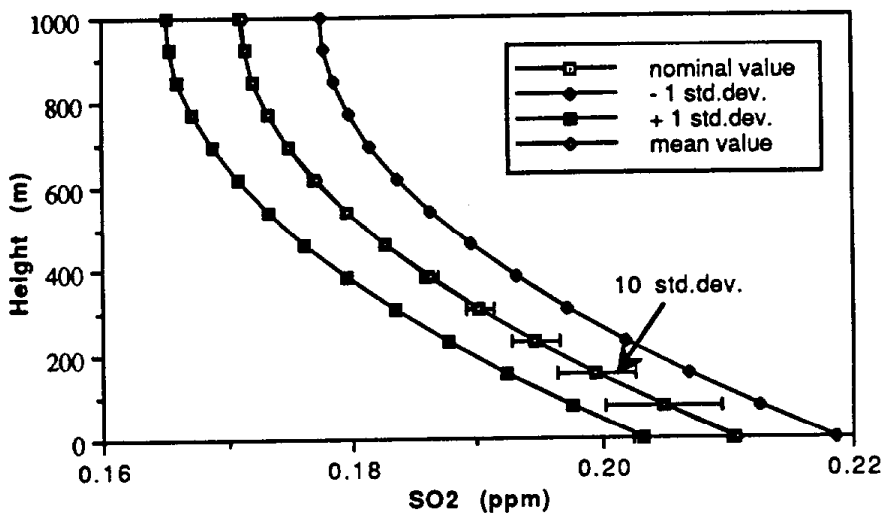


Figure 5.10 Concentration profile of SO_2 at time = 19 by Monte Carlo for $\sigma(D) = 0$ %, $\sigma(k) = 0$ %, $\sigma(v_g) = 20$ %.

5.7 Conclusions

There are several key conclusions to be drawn from the preceeding analyses. The first is that by using the new Hammersley-Wozniakowsky sampling procedure it is possible to significantly reduce the number of model simulations needed to develop stable estimates of the effects of data errors. One of the most critical findings is the fact that just sampling from the extremes of the distributions considerably overestimates the likely ranges in the predicted concentration variables. Without a doubt the most important finding was the fact that uncertainties in the predicted concentrations do not grow exponentially with time. In fact the uncertainties in the predictions arising from data errors decayed at the end of each diurnal cycle. Considering the magnitude of the uncertainties in the predictions arising from the data errors it is possible to conclude that just because there may be an error in one of the data inputs it does not imply a similar level of uncertainty in the predictions of the airshed model.