Mathematical Modeling of Air Pollution Dynamics by Parallel Computation

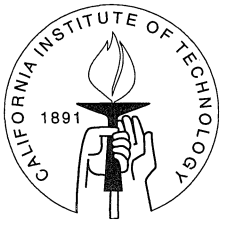
Thesis by

Donald Dabdub

In Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy



California Institute of Technology

Pasadena, California

1996

(Submitted September 5, 1995)

Dedicada a mis abuelos Adrián y Teófilo

To my grandfathers Adrián and Teófilo

Acknowledgements

I wish to express my gratitude and deep appreciacition to my advisor Prof. John Seinfeld for his support, thoughtfulness, and for providing trust and freedom in conducting this research.

The comments, challenges, and advice given to me by Dr. Rick Flagan and Dr. K. Mani Chandy have been extremely helpful. I thank them for their time, interest and encouragement. I would like thank Dr. Herb Keller, Dr. Eric van de Velde, Dr. George Gavalas, Dr. William Schiesser (Lehigh University) and Dr. Marco Duran (Exxon Research) for their superb teachings.

I have been fortunate to have participated in animated discussions with my friends Rajit Manohar, Suresha Guptha, Sharon Brunett and Carl Kesselman. They shared with me their passion to design faster, shorter, and more efficient computer tasks.

I have benefited greatly from a number of friendships that I formed during my stay at Caltech. I am particularly thankful to my officemate Jean Andino, my computer-lab-mate Patrick Chuang, and my roommate Jeff Lindley for their moral support during long work hours.

The Caltech staff have been extremely helpful during my research work. Especially I want to thank Marta Goodman, April Olson and Kathy Lewis for their capable assistance.

I gratefully acknowledge the various funding agencies which have supported my work: IBM Corporation, United States Environmental Protection Agency, and The Center for Air Quality Analysis.

Finally, I thank my family for their encouragement and their blessings.

Gracias!

Abstract

The use of massively parallel computers provides an avenue to overcome the computational requirements in the study of atmospheric chemical dynamics. General considerations on parallel implementation of air quality models are outlined including domain decomposition strategies, algorithm evaluation and design, portability, modularity, and buffering techniques used in I/O operations. Results are given for the implementation of the CIT urban air pollution model on distributed memory multiple instruction / multiple data (MIMD) machines ranging from a cluster of workstations to a 512 node Intel Paragon.

The central challenge in developing a parallel air pollution model is the implementation of the chemistry and transport operators used in the solution of the atmospheric reaction-diffusion equation. The chemistry operator is generally the most computationally intensive step in atmospheric air quality models. A new method based on Richardson extrapolation to solve the chemical kinetics is presented. The transport operator is the most challenging to solve numerically. Because of its hyperbolic nature non-physical oscillations and/or negative concentrations appear near steep gradient regions of the solution. Six algorithms for solving the advection equation are compared to determine their suitability for use in parallel photochemical air quality models. Four algorithms for filtering the numerical noise produced when solving the advection equation are also compared.

A speed-up factor of 94.9 has been measured when the I/O, transport, and chemistry portions of the model are performed in parallel. This work provides the computational infrastructure required to incorporate new physico-chemical phenomena in the next generation of urban- or regional-scale air quality models.

Finally, the SARMAP model is used to model the San Joaquin Valley of California. SARMAP is the updated version of RADM. It can be considered a state-of-the-art regional air pollution model. Like the CIT model, SARMAP incorporates the fol-

lowing atmospheric phenomena: gas-phase chemistry, advection and diffusion. In addition, SARMAP incorporates aqueous-phase chemistry and transport through cumulus clouds. Sensitivity studies performed show a significant dependence of ozone model predictions on boundary conditions.

Contents

List of Figures

List of Tables

Chapter 1

Air Quality Modeling on Massively Parallel Computers

Donald Dabdub and John H. Seinfeld.

Atmospheric Environment, **28**, (1994) 1679–1687.

The use of massively parallel computers provides an avenue to overcome the computational requirements of air quality modeling. General considerations on parallel implementation of air quality models are outlined including domain decomposition. The implementation of the CIT urban photochemical model on the Intel Touchstone Delta, a distributed memory multiple instruction / multiple data (MIMD) machine is described. When both the transport and chemistry portions of the model are parallelized, a speed-up of about 30 is achieved using 256 processors.

1.1 Introduction

Air quality models are essential tools in the understanding of pollutant dynamics in the atmosphere. In recent years, our understanding of the scientific foundations of the chemical and physical phenomena occurring in the atmosphere has continued to expand. We are able to construct comprehensive models that describe the dynamics of air pollution. The inherent complexity and nonlinearity of the governing equations has made air quality modeling a computational "Grand Challenge" (Levin, 1989). Currently, air quality modeling is most often performed on sequential computers.

Computational constraints have always been a limiting factor in the amount of physics and chemistry one can include in air quality models (AQMs). For example, particulate formation processes are not currently incorporated in most models due to the significant time demands of the aerosol phase computations (see, for example, Pilinis and Seinfeld, 1988). Phenomena occurring in the sub-grid scale are also ignored or crudely represented by current AQMs. The use of parallel computers provides an avenue to overcome the computational requirements of air quality modeling.

The work reported here has as a major goal to lay the foundation to implement air quality models on parallel computers. To accomplish this goal it is necessary to study and compare different approaches to distribute the computational work among the available nodes. It is necessary to test, compare, and evaluate current numerical schemes employed in the solution of AQMs. Implications of parallel computation on restructuring of the input and output data must also be examined.

1.2 Computational Breakdown of Current AQMs

The governing equation of three-dimensional Eulerian AQMs is the atmospheric diffusion equation:

$$\frac{\partial c_i}{\partial t} + \mathbf{u} \cdot \nabla c_i = \nabla \cdot (\mathbf{K} \cdot \nabla c_i) + R_i(\mathbf{c}, T) + S_i(\mathbf{x}, t) \tag{1.1}$$

where c_i are the elements of the concentration vector $\mathbf{c}$, t is time, $\mathbf{x} = (x, y, z)$, $\mathbf{u} = (u, v, w)$ is the advective flow field, $\mathbf{K}$ is the eddy diffusivity tensor, R_i is the chemical production of species i, T is the temperature, and S_i is the source rate of i.

The different chemical and physical processes that contain inherently wide variations in their time scales pose the major challenge in constructing numerical methods to solve equation (1.1). Operator splitting methods have been developed and refined for the solution of AQMs (McRae *et al.*, 1982a). Splitting methods provide a numerical approach that is both accurate and economical.

The basic idea of the splitting process is the sequential use of operators, $\mathcal{L}$, that govern the different phenomena. Horizontal transport is described by:

$$\mathcal{L}_H c_i = \frac{\partial c_i}{\partial t} = -\nabla_H \cdot \mathbf{u} c_i + \nabla_H \cdot \mathbf{K} \nabla_H c_i, \tag{1.2}$$

where H represents the (x, y) plane. Vertical transport is described by

$$\mathcal{L}_z c_i = \frac{\partial c_i}{\partial t} = -\frac{\partial}{\partial z}(w c_i) + \frac{\partial}{\partial z}(K_{zz} \frac{\partial c_i}{\partial z}). \tag{1.3}$$

Finally, chemistry and emissions are described by:

$$\mathcal{L}_c c_i = \frac{\partial c_i}{\partial t} = R_i(\mathbf{c}, T) + S_i(\mathbf{x}, t). \tag{1.4}$$

In some AQMs $\mathcal{L}_H$, the horizontal transport operator, is decomposed into two separate operators, $\mathcal{L}_x$ and $\mathcal{L}_y$, describing the transport in the x and y directions, respectively. In addition, current AQMs often combine $\mathcal{L}_c$ and $\mathcal{L}_z$ into a single operator $\mathcal{L}_{cz}$ that performs the chemistry and vertical transport computations simultaneously.

Table 1.1 summarizes several existing AQMs from the point of view of their computational characteristics. UAM and CIT are urban scale air quality models. Urban models typically have a vertical domain extending up to about 2 km, as opposed to regional models that extend up to about 10 kmin order to treat vertical redistribution of species above the planetary boundary layer. RADM, ADOM and STEM-II are regional acid deposition and oxidant models that treat gas- and aqueous-phase chem-

istry. They have been applied primarily to simulations of acid deposition in Eastern North America and central Japan. ROM is a regional oxidant model designed to simulate ozone formation and transport over the eastern United States.

The solution of the atmospheric diffusion equation in the operator splitting framework is obtained from the following sequence:

$$\mathbf{c}^{t+2\Delta t} = \mathcal{L}_x^{\Delta t}\mathcal{L}_y^{\Delta t}\mathcal{L}_{cz}^{2\Delta t}\mathcal{L}_y^{\Delta t}\mathcal{L}_x^{\Delta t}\mathbf{c}^t. \tag{1.5}$$

The amounts of time spent computing the solutions of the different operators of the CIT model, for example, are: chemistry 85.2%, horizontal transport 5.4%, "other" 9.4%. The chemistry loop of the code contains the vertical transport computations since they are coupled in the $\mathcal{L}_{cz}$ operator. In addition, the chemistry loop also contains the deposition computations for all the species within all air columns, as well as the vertical transport routines. All these computations make up approximately 90% of the code. "Other" represents the reading and writing of data, the initialization of the model, and other minor computations. It is expected that other three-dimensional AQMs have approximately the same computational breakdown.

Most of the computer time involved in solving the core equations of urban and regional scale photochemical models is consumed by the $\mathcal{L}_{cz}$ operator. The "chemistry" part of the operator consists of solving a coupled system of stiff, nonlinear ordinary differential equations as described by equation (1.4). Classical methods for solving systems of stiff ordinary differential equations, such as Gear's method, are not of practical use in AQMs. The reason is due not to their accuracy, but to the small time steps required, the inversion of large matrices, and/or the repeated solution of large sets of nonlinear equations. A comparison of different numerical schemes to perform the integration of the chemistry in AQMs can be found in Hov *et al.* (1991) and Odman *et al.* (1992).

The solution of the advection-diffusion equation (1.2) requires 5.4% of the computational time in the CIT model. When solving this equation numerically, depending on the scheme used, both the amplitudes and the phases of different Fourier

Table 1.1: Aspects of current photochemical and acid deposition models.

Model	**Vertical Resolution**	**Horizontal Resolution**	**Gas-Phase Chemistry**	**Aqueous-Phase Chemistry**	**Transport**
UAM	4 layers up to 1.5 km	4 km × 4 km to 10 km × 10 km	71 reactions 30 species	Not treated	3D wind field.
CIT Mc Rae and Seinfeld (1982b) Harley *et al.* (1993)	5 layers up to 1.5 km	5 km × 5 km	71 reactions 30 species	Not treated	3D wind field.
ROM Lamb (1982) Schere and Wayland (1991)	3 layers up to 2 km	18 km × 18 km	71 reactions 30 species	Not treated	3D wind field with vertical transport through cumulus clouds
RADM-II Chang *et al.* (1987)	15 layers up to 10 km	80 km × 80 km	104 reactions 58 species	42 equilibria 5 rxn for SO_2 oxidation	3D wind field with vertical transport through cumulus clouds
STEM-II Carmichael *et al.* (1986)	10 to 14 layers up to 6 km	10 km × 10 km to 56 km × 56 km	112 reactions 53 species	26 equilibria 30 rxn for SO_2 and NO_x oxidation	3D wind field with vertical transport through clouds
ADOM Venkatram *et al.* (1988)	12 layers up to 10 km	60 km × 60 km to 120 km × 120km	∼100 reactions ∼ 50 species	25 reactions 13 species	3D wind field with vertical transport through clouds

components of the solution will be altered. This produces so-called numerical diffusion and dispersion. This is a classic problem in computational fluid dynamics, for which a large number of specialized numerical techniques have been developed (Rood, 1987). Many have been tested and compared to determine their applicability to AQMs (Chock *et al.*, 1983, 1985, 1991). The numerical method currently implemented in the CIT model, for example, employs a fourth order (space) Chapeau-function based finite element scheme as part of an implicit procedure to solve the advection part of $\mathcal{L}_H$. Following the advection step, a nonlinear noise filter is used to remove most of the computational noise generated by the scheme (Forester, 1977). Finally, the diffusion step is computed with an explicit second-order finite difference scheme (McRae *et al.*, 1982a). In brief, the challenges of solving equation (1.2) relate largely to computational accuracy.

The analysis of where computational effort is expended shows that a parallel implementation of an AQM should start by focusing on performing the chemistry integrations simultaneously on different processors.

1.3 Parallel Architectures

Traditionally, parallel architectures can be classified into two broad categories: SIMD and MIMD. Each has different programming approaches.

SIMD architectures, denoting single instruction/multiple data, execute the exact same instruction on different sets of data simultaneously. SIMD machines are well suited for problems that primarily require the manipulating of large matrices.

MIMD architectures, for multiple instruction/multiple data, can execute different instructions on different sets of data simultaneously. The way that the different processors communicate determines the different flavors of MIMD. In a shared-memory MIMD computer all the processors have access to a common memory. Shared-memory MIMD machines are relatively easy to program, but they are limited by scalability and they might present cache-coherency problems. On the other hand, in a distributed-memory MIMD machine every processor contains its own local memory. Distributed-

memory MIMD machines are also known as multicomputers. Intel's Touchstone Delta is one of the newest and fastest multicomputers. A network of workstations optically interconnected qualifies also as a parallel multicomputer.

It is not currently known with full certainty which architecture provides the best environment for air quality models. The issue of exploiting the different advantages of a given parallel architecture still remains to be studied. Carmichael *et al.* (1989) have studied the solution of transport/chemistry calculations on SIMD machines. They have observed that simple (4 species) chemical mechanisms are well suited for SIMD machines. However, the implementation of transport algorithms efficiently on SIMD machines leaves much to be desired.

Pai and Tsang (1992) have used different shared-memory MIMD machines to study common time-splitting finite difference or finite element schemes used in air pollution modeling. Specifically, they simulated turbulent diffusion in convective boundary layers. The highest speed-up reported is 13.11 using a Sequent Symmetry S81 machine.

Shin and Carmichael (1992) have parallelized the STEM-II air pollution model. They used a shared-memory ALLIANT FX/8. Their work focused on parallelizing the chemistry portions of the model. The speed up reported reached to 2.5. Furthermore, they mention that the efficient use of vectorization would require alternative algorithms.

We report here on the use of distributed-memory MIMD machine architectures for air quality modeling. We have implemented the CIT model on the Intel Touchstone Delta (Intel, 1991). The Delta contains 570 nodes: 528 numeric nodes, 34 mass storage nodes, 2 gateway nodes, and 6 service nodes. The nodes are interconnected as a two-dimensional mesh. A maximum configuration would provide 10 Gigabytes of distributed main memory and 200 Gigabytes of online storage[1]. The 528 numeric nodes are based on the 64-bit Intel i860 microprocessor. The i860 has a peak speed of

[1]Currently, three-dimensional air quality models considering only gas-phase phenomena do not impose extreme memory requirements, nor extreme storage space requirements. However, with the implementation of aqueous phase and/or aerosol phase, the memory and storage requirements would increase drastically.

60 double-precision MFLOPS and 80 single-precision MFLOPS. Each numeric node includes 16M bytes of parity-check DRAM, a 4K byte instruction cache, and an 8K byte data cache.

The programming was done in FORTRAN 77 using NX version 1.4 as the extended communication library used for message passing. The issue of software portability requires special attention to different aspects of the programming. First, only the simple synchronous send and synchronous receive routines should be used. By avoiding the use of more sophisticated message passing routines, a degree of freedom is gained on code portability. That is, porting of the parallel code to other message passing routines such as EXPRESS, LINDA, or PVM can be achieved by replacing the low level synchronous sends and receives with the appropriate library call. Furthermore, the use of NX as the underlying message passing library makes the code executable on other parallel computers commercially available with no changes. Specifically, the code has been run successfully on the Intel GAMMA, the IPSC and IPSC2. Second, one should write general purpose send/receive routines. By doing so, these general send/receive routines can call any appropriate communication protocol dependent send/receives. Third, the code should be independent of the number of nodes available. The number of nodes available should be determined at execution time or given as a parameter so that no further code modifications are needed. Finally, the use of parallel file systems should be avoided from the point of view of portability (not performance) since they are highly architecture dependent.

The use of MIMD machines provides a promising approach to AQM calculations. MIMD provides a friendly environment for SPMD (single process multiple data) and for MPMD (multiple process multiple data) programming methodologies. The chemistry computations in AQMs follow a SPMD approach, while the transport computations, as described below, inherently follow a MPMD approach. By definition, SIMD architectures are not suitable to MPMD programming. Furthermore, the usage of distributed-memory MIMD opens the possibility of porting the code to a network of workstations, which might be more readily available than a parallel supercomputer. In short, MIMD seems to be the most promising architecture to perform AQM com-

putations.

1.4 Key Issues in the Parallel Implementation of AQMs

The first step towards parallelizing a code such as an AQM is to perform a detailed profile of the code. A code profile, as discussed earlier, shows the computational breakdown of the code, that is, the computational time spent in each routine for an entire run. In principle, the computationally intensive parts are the desired subroutines to be performed in parallel. For the case of comprehensive AQMs, as noted above, the most computationally intensive routine is the chemistry integration. The amount of time spent here ranges from 75% to 95% of the total computing time.

The next issue of concern is that of data dependency. The AQM chemistry portion does not present any difficulty on data dependency. All the data needed to compute the vector of species concentrations after the chemical changes of the corresponding time step are local. The results of the chemistry integration at a particular grid point do not depend on the concentrations at other grid points. On the other hand, transport computations, regardless of the numerical scheme used, are, in principle, dependent on data located at neighboring grid points. Communications among the nodes is therefore imperative. The implementation of node communication is dependent on the transport numerical scheme used, as well as on the decomposition of the domain implemented.

Speed-up is a common measure of performance gain from a parallel environment. It is the factor denoting how many times faster the parallel version of the code runs in comparison to the sequential version. Indeed, it is defined as the ratio of the time required to complete a job using one processor to the time required to complete the same job with N processors. The maximum speed-up is limited by the fraction of that job that is performed in parallel, p. An ideal speed-up, S, achieved with

parallelization is described by Amdahl's law:

$$S = \frac{1}{(1-p) + p/N}. \tag{1.6}$$

The speed-up is ideal in the sense that it does not account for the time taken to send/receive messages. Increasing p by a small amount at a point where a significant amount of the code is already parallelized, say for $p > 0.96$, produces substantial additional speed-up.

1.4.1 Domain Decomposition

The objective of domain decomposition is to distribute the computational load as evenly as possible among the available nodes. The different ways to decompose the three-dimensional spatial domain present several questions to be addressed for an effective parallel AQM implementation. There are two main approaches to this matter.

The first approach to decompose the domain is called dynamical domain decomposition. One starts with a number of tasks to be performed. A so called "master" node is in charge of sending one of those tasks to a node that is idle at the moment. The idle node receives the message and some appropriate data needed to perform the task. The appropriate data might come from the master node and/or from other nodes that should be notified of such requests. After the task is completed, the node sends the results to the master node, or to other nodes, and/or keeps it stored in its own memory. The node also signals the master that it has finished the task. The process is continued in this way until all the desired tasks are performed.

The other approach to distribute the computational load is to set a predetermined decomposition. In this case, the master sends a specific number of tasks to each node. The tasks are distributed at once among the available nodes. The nodes, when having completed their collection of tasks, signal the master that they have done so. The advantage of this technique arises when the programmer is able to set an approximately evenly distributed set of tasks. The technique also facilitates debugging of the code, since the programmer knows what task is in what node.

Furthermore, there is a decrease in overhead work since the communication between the nodes and the master node is minimized.

Another issue to be considered while determining the domain decomposition to be used is the dimensionality of the decomposition. One has to decide on performing either a one-dimensional or a two-dimensional decomposition. Fox *et al.* (1988) discuss the implications of the dimensionality of the decomposition. In the case of AQMs the chemistry computations are independent of the dimensionality of the decomposition. However, the dimensionality of the the decomposition will likely affect the transport computations.

Figure 1.1 shows the different domain decompositions used as a function of the number of slave nodes available for the simulation of the South Coast Air Basin using the CIT model. The rectangle represents the x-y projection of the master data storage grid of the model. The grid is currently subdivided into 80 cells in the x-direction, 30 cells in the y-direction, and 5 layers. An individual cell in the figure represents a 5×5 km^2 area. The shaded section in the 1-node case of Figure 1.1 represents the computational region within the overall region. In the 3-node and 7-node decompositions of Figure 1.1 the alternations of gray shades indicate the boundaries of the data as they are distributed among the available slave nodes. As can be seen, we have implemented a domain decomposition as close to a purely one-dimensional decomposition as possible. The reason to stay close to the one-dimensional case is discussed subsequently in the Horizontal Transport sub-section.

The rule followed to determine the domain decomposition is to have an approximately equal number of vertical columns sent to each node. In this manner, the load is well balanced. For chemistry implementation purposes only, the efficiency of the implementation is not dependent on the dimensionality of the decomposition.

1.4.2 Chemistry

As mentioned before, chemistry computations are the major computational load of typical AQMs. Their parallel implementation is rather simple since it does not in-

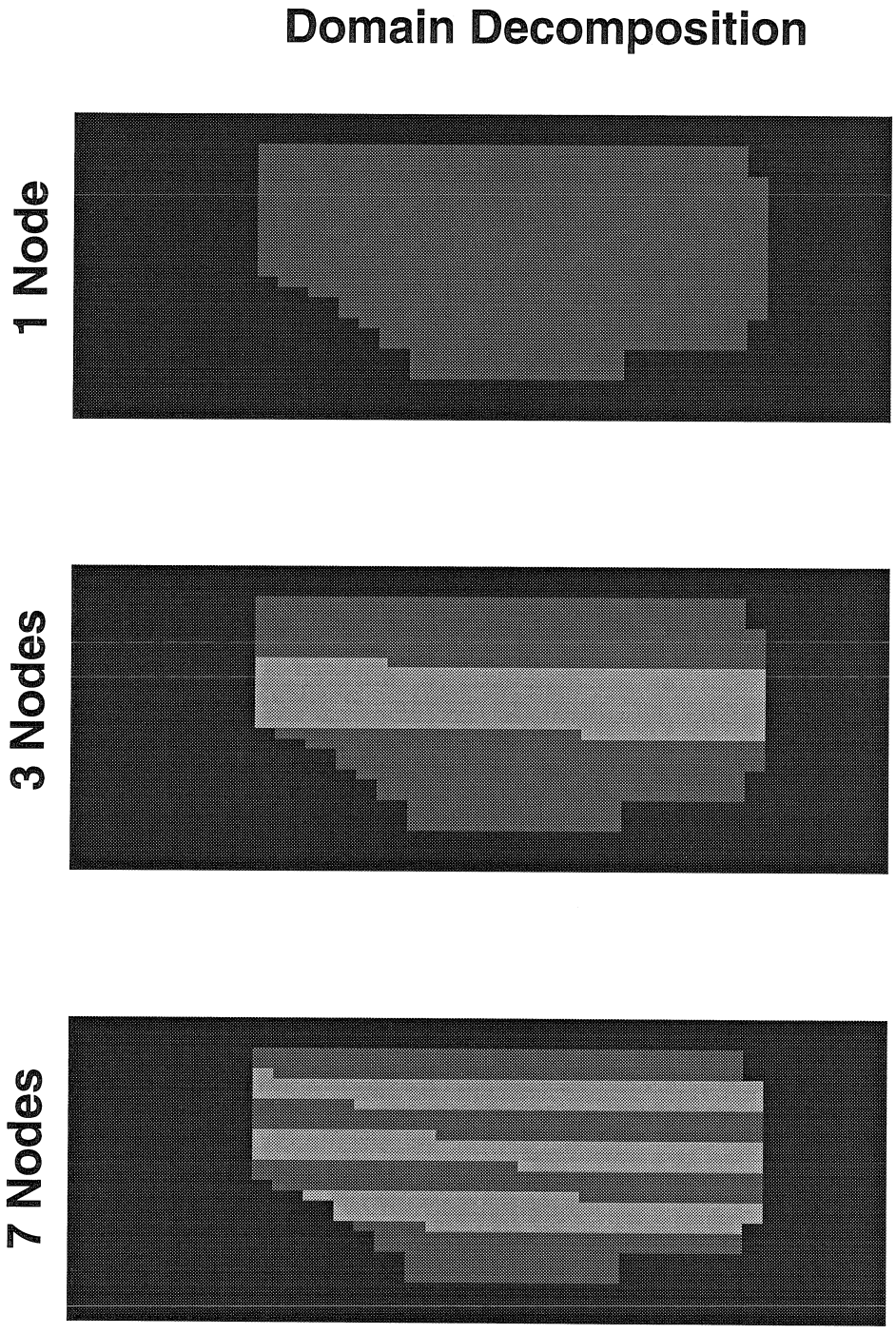


Figure 1.1: Domain decomposition of the South Coast Air Basin region for different numbers of slave nodes. The dark rectangle represents the overall domain. The gray shaded area represent the computational domain. Different shades of gray represent sub-sections of the computational domain to be distributed to a specific node.

volve communication among the "slave" nodes. The chemistry integrator used in the CIT model is an implicit, hybrid, asymptotic, exponential scheme developed by Young and Boris (1977). Parallel implementations of the chemistry integrator are a function neither of the numerical scheme used for the integration nor of the particular photochemical mechanism. Thus, parallel implementation of the chemistry does not interfere with the modularity of the code to any degree.

An effective approach is to send a collection of vertical columns to a node, using a predetermined domain decomposition. The programmer can set a predetermined decomposition rather easily in an AQM. Assuming that the time it takes to integrate any vertical column of cells is approximately the same, then one tries to send the same number of vertical columns to each slave node available, so that all the nodes finish their tasks at approximately the same time. The number of columns to be sent to each node should be computed dynamically. It is approximately equal, of course, to the total number of columns divided by the number of slaves nodes available.

Figure 1.2 shows the time to perform a 24-hour standard simulation of the South Coast Air Basin of California using the CIT model. The points plotted correspond to 2, 4, 8, 16, 32, 64, 128 and 256 nodes of the Intel Delta. The distance between the ideal and measured curves in Figure 1.2 represents the time spent in the communication between the master node and the slave nodes, as well as idle time among the slave nodes. Figure 1.2 shows that the time continues to decrease as more and more nodes are used. This is the expected behavior in the small number of processors regime. Sometimes, however, an increase in time actually occurs when the number of processors is increased beyond a certain point. This phenomenon, if present, occurs at the high number of processors regime (i.e., the massively parallel regime). The reason is that the master node has so many slaves to manage that the overall productivity of the group as a whole decreases. Using 256 nodes, the speed-up obtained by parallelizing the $\mathcal{L}_{cz}$ operator is 13.9. That is, a 24-hour simulation takes 46 min to complete.

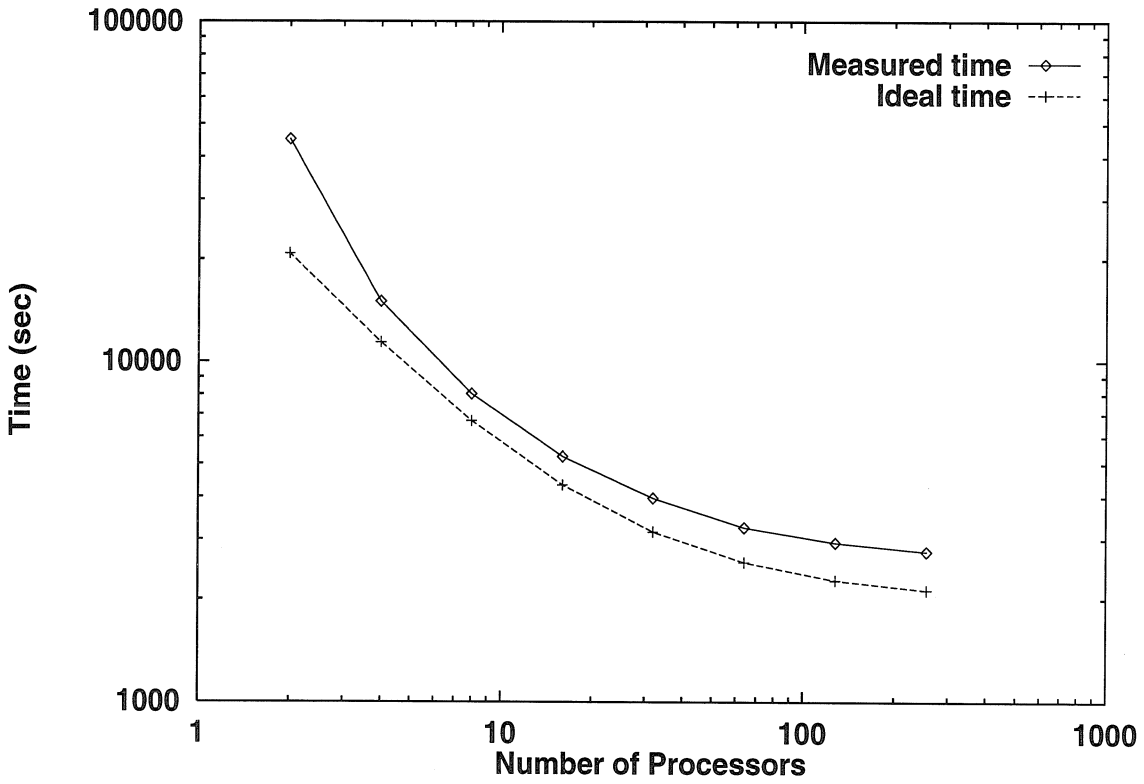


Figure 1.2: Execution time as a function of number of nodes when parallelizing the chemistry loop of the model. Ideal time presented is calculated from Amdahl's law. Time reported corresponds to a 24-hour standard simulation of the South Coast Air Basin.

1.4.3 Vertical Transport

In the CIT model vertical transport is part of the $\mathcal{L}_{cz}$ operator. In the chemistry implementations in parallel the entire air column was sent to the nodes to be integrated. Therefore, by sending the rest of the data needed, such as the z component of the advective flow, one can perform the vertical transport computations with no further complications. In the same way, deposition velocity calculations and other computations included inside the "chemistry" loop of the code should be performed in parallel.

If the chemistry operator is not coupled with the vertical transport, parallel performance is not greatly affected . Usually the percentage of the time spent in computing vertical transport is quite small. Particularly, in the CIT model it is less than 3% and is included in the "other" category. According to Amdahl's law, that 3% of code would, however, be crucial if most of the rest of the code could be implemented in parallel.

1.4.4 Horizontal Transport

When the chemistry integration has been implemented successfully, as well as other computations such as the vertical transport and deposition, significant speed-ups are obtained. At this point, any implementation of other sections of the code will have a great impact on increasing speed-ups in accordance with Amdahl's law. The effect of implementing in parallel a constant small percentage of the code is proportional to the percentage of the code already parallelized. In particular, the CIT model spends 5.4% of its time performing the horizontal transport computations. This 5.4% would increase the speed-up of the code if implemented in parallel, and, the value of the increased speed-up would be proportional to the percentage of the code already parallelized as reflected in Amdahl's law.

The implementation of transport computations requires communication among the slave nodes. The communication needed for optimal performance is dependent on the numerical scheme used. The scheme determines the number of neighboring

grids used to compute the values of the species concentration to advance the next transport step. An optimal performance implementation of horizontal transport can use the standard technique known as extended arrays. Extended array techniques consist of having the nodes send the values of species concentrations contained in the boundaries of their domains. The nodes would also receive the values of the concentration boundaries of their neighboring nodes.

The horizontal transport operator in the CIT model is decomposed into two separate operators $\mathcal{L}_x$ and $\mathcal{L}_y$. Our implementation of the transport computations did not make use of extended arrays. Thus, optimal performance was sacrificed for the sake of maintaining modularity.

We can illustrate the implementation of the transport computations with the approach we have used. For the $\mathcal{L}_x$ operator, the node containing the leftmost row of the computational grid is the one dedicated to perform the transport computations for that particular row. The assigned node requests data from the other nodes that contain data of that particular row. Note that data might be needed from more than one node. The other nodes send the data to the assigned node. All the data required to perform transport computations are now in the assigned node. At this point, any algorithm can be used to compute the concentrations at the next transport step. All the communications within rows, as well as the transport computations, are carried out simultaneously at all the assigned nodes. Finally, all the assigned nodes must send the computed concentrations to the appropriate neighboring nodes.

The choice of the node as the place where computations are to be performed is arbitrary. For this idea to be implemented successfully, a unique message identifier must be set for each message going back and forth to the assigned node. This method, however, can be taken a step further. For instance, instead of having only one assigned node to compute the next transport step of the all the species in all 5 vertical layers, one can assign 5 nodes to compute simultaneously all the species on each layer.

It is now clear that a one-dimensional domain decomposition helps to reduce the message passing when performing transport computations in the x-direction and using a small number of nodes. The y-direction transport is not affected. In comparison,

the use of a two-dimensional decomposition results in a greater number of messages passed in the x-and y-directions. This is the case even when computing with a small number of slave nodes.

The transport in the y-direction was implemented using the same basic idea. All the y-column transport computations are to be performed at an arbitrarily assigned node. After receiving data from other columns, and performing the y-transport computations, the assigned node sends back the vector of species concentrations to the corresponding nodes. Even though this approach is not the optimal implementation to perform the horizontal transport computations, it is independent of the numerical scheme used.

The implementation of the horizontal transport in parallel increases the percentage of parallelized code by 5.4%. Since a significant percentage of the code has been already implemented in parallel, the speed-up increase is quite noticeable. Figure 1.3 shows that computing time continues to decrease as the number of nodes increases up to 256. Furthermore, it shows that the distance between the measured time and the ideal time is quite dependent on the number of nodes. Figure 1.3 shows a greater discrepancy between the measured times and ideal times when the chemistry and transport are both performed in parallel than Figure 1.2 when only chemistry is parallelized. The reason is that the current implementation of horizontal transport sacrifices optimal transport implementation to gain modularity in the code. However, Figure 1.4 shows that the parallel implementation of the transport makes the code run faster as long as 8 or more nodes are used. The time saved is proportional to the number of nodes used, as expected. When using only 4 nodes, the parallel transport implementation actually runs slower than when only the chemistry is implemented in parallel, a direct consequence of the overhead of intensive message passing that occurs to perform the transport calculations. As more nodes become available, the increase in performance overshadows such overhead.

When implementing the transport in parallel a greater speed-up is obtained than when only the chemistry part of the code is parallelized. The best speed-up measured is 28.47 using 256 nodes. That is, a 24-hour run takes less than 23 minutes to compute.

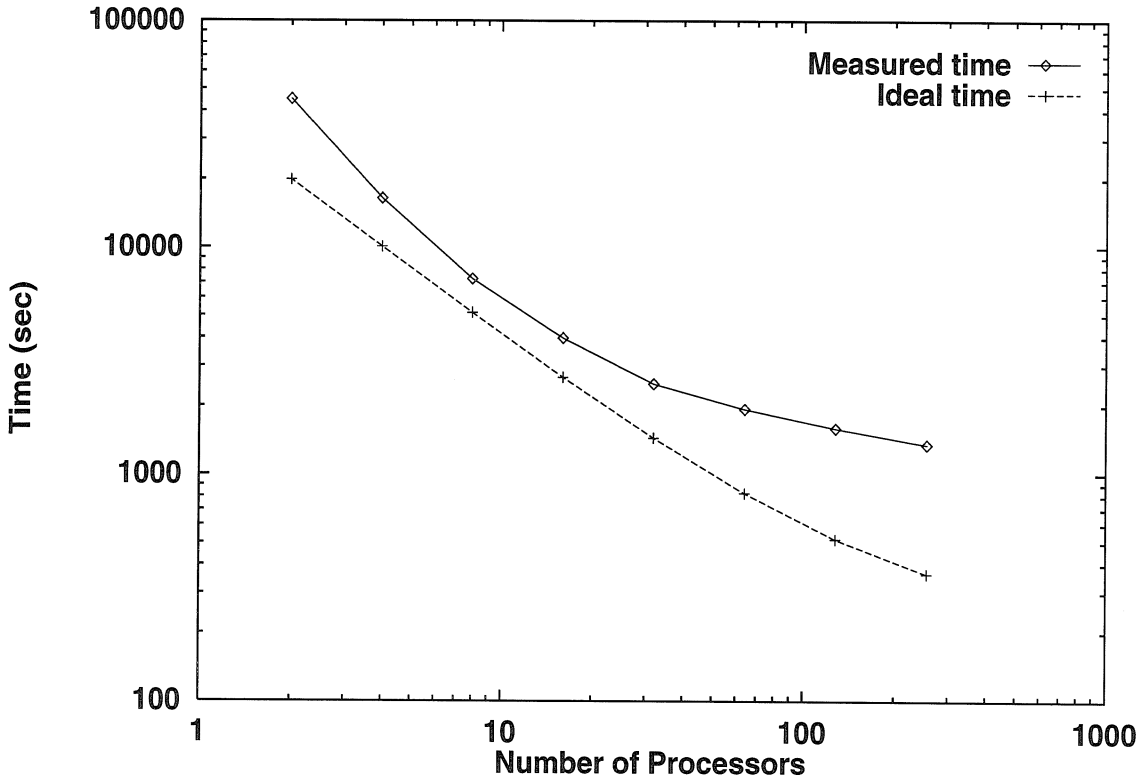


Figure 1.3: Execution time as a function of number of nodes when parallelizing the chemistry loop and the transport equation solver of the model. Ideal time presented is calculated from Amdahl's law. Time reported corresponds to a 24-hour standard simulation of the South Coast Air Basin.

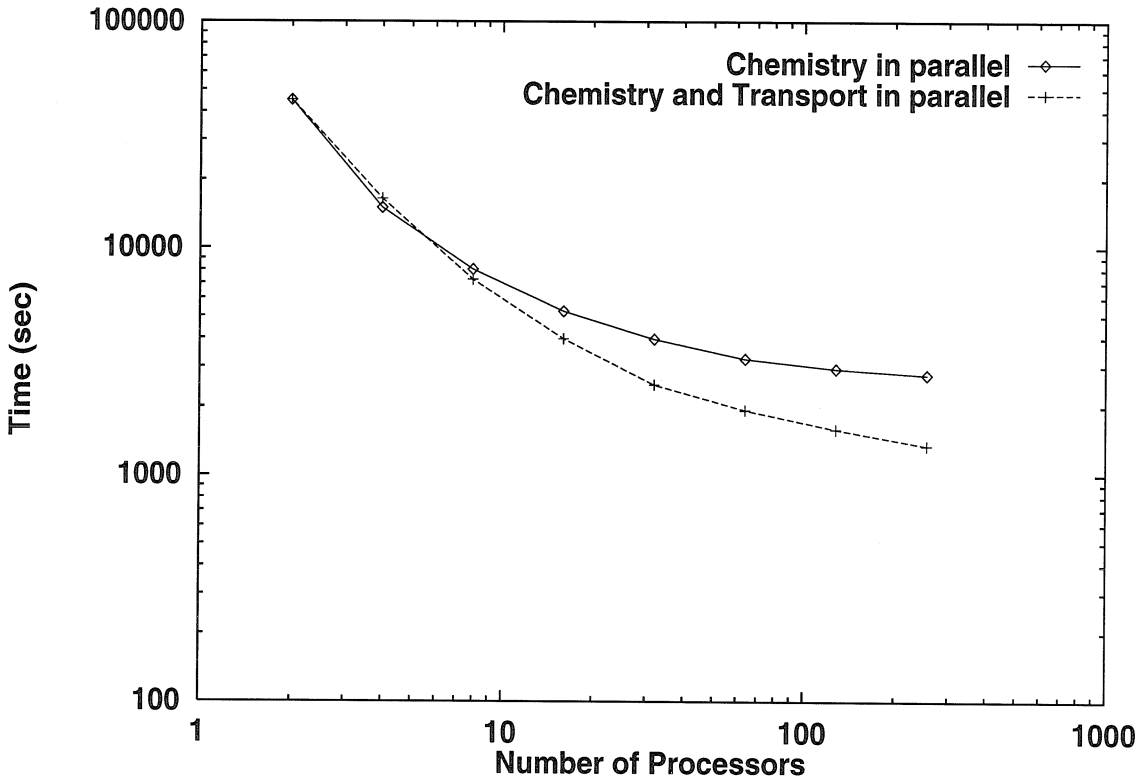


Figure 1.4: Execution time as a function of number of nodes. One set of data corresponds to the parallelization of the chemistry loop only. The other data set corresponds to the model having the chemistry loop and the transport equation solver implemented in parallel. Time reported corresponds to a 24-hour standard simulation of the South Coast Air Basin.

Figure 1.5 shows the computational time of the CIT model on different systems. The times reported correspond to simulating a 3-day air pollution episode in the South Coast Air Basin.

1.5 Conclusions

This work presents the implementation of the CIT photochemical air quality model on the Intel Touchstone Delta. It has been found that the use of MIMD computer architectures provides an excellent environment for air quality models.

The implementation of the chemistry section of the code involves a simple host to node communication pattern. The parallel implementation of the chemistry is independent of the numerical scheme and the photochemical mechanism used. When the chemistry portion of the CIT model is parallelized, a speed-up factor of 13.9 is achieved using 256 nodes.

Implementation of the horizontal transport section of the code requires more careful programming than for the chemistry since it inherently contains communication among nodes. The parallel implementation of the horizontal transport is dependent on the order of the numerical scheme, if extended array techniques are used. To obtain independence and keep the model modular with respect to the transport solver, a small performance price must be paid. When the horizontal transport portion of the CIT model is parallelized in addition to the chemistry, a speed-up factor of 28.47 is achieved using 256 nodes.

Parallel computing is a powerful tool for air pollution modeling. By significantly reducing the computing time, it allows more detailed treatment of the dynamics of the atmosphere and it provides the numerical power necessary to meet the needs of future generation AQMs.

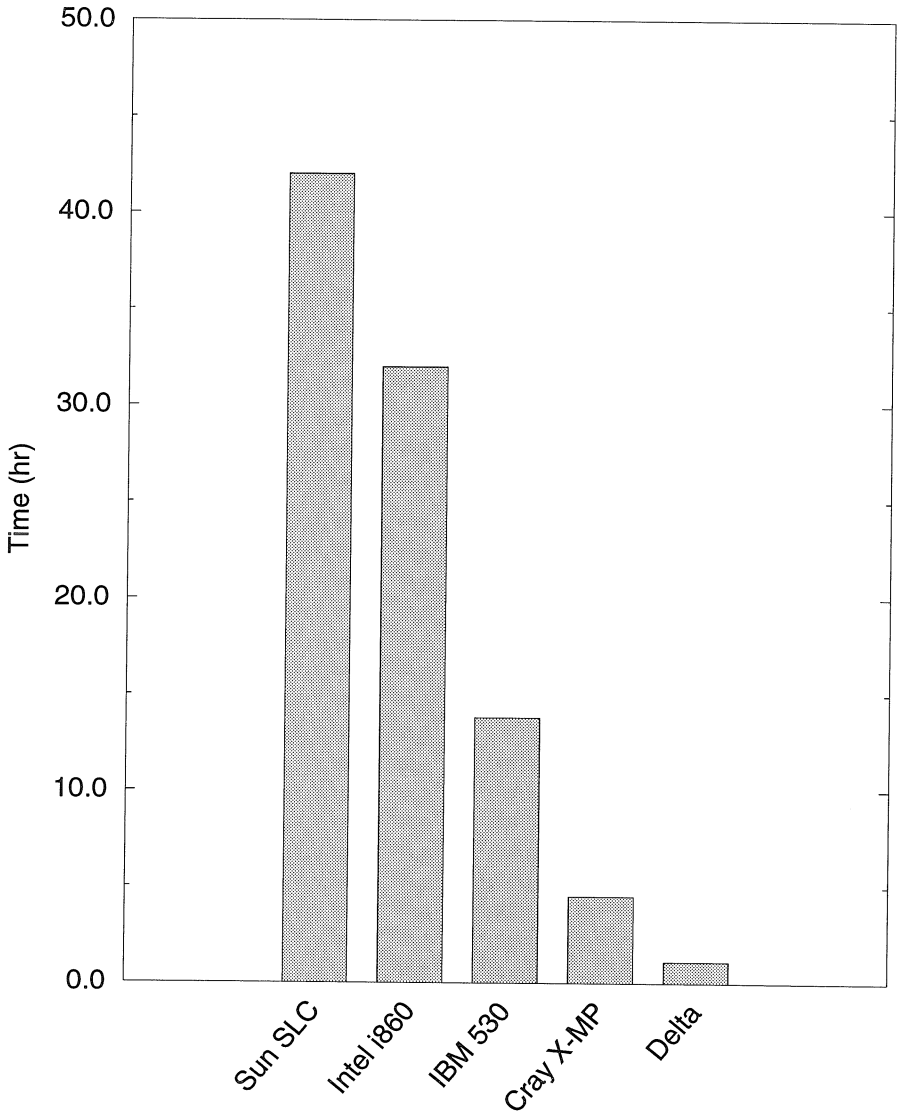


Figure 1.5: Execution time for different computing platforms. Time reported corresponds to a 3-day simulation of the South Coast Air Basin. Delta result corresponds to the model having the chemistry loop and the transport equation solver implemented in parallel on the Intel Touchstone Delta using 256 processors.

References

Carmichael G. R., Peters L. K. and Kitada T. (1986) A second generation model for regional-scale transport/chemistry/deposition. *Atmospheric Environment* **20,** 173-188.

Carmichael G. R., Cohen D. M., Cho S. Y. and Oguztuzun, M. H. (1989) Coupled transport-chemistry calculations on the massively parallel processor computer. *Comput. Ch. E.* **13,** 1065-1073.

Chang J. S., Brost R. A., Isaksen I. S. A., Madronich S., Middleton P., Stockwell W. R. and Walcek C. J. (1987) A three-dimensional Eulerian acid deposition model: physical concepts and formulation. *J. Geophys. Res.* **92,** 14681-14700.

Chock D. P. and Dunker A. M. (1983) A comparison of numerical methods for solving the advection equation. *Atmospheric Environment* **17,** 11-24.

Chock D. P. (1985) A comparison of numerical methods for solving the advection equation—II. *Atmospheric Environment* **19,** 571-586.

Chock D. P. (1991) A comparison of numerical methods for solving the advection equation—III. *Atmospheric Environment* **25A,** 853-871.

Forester C. K. (1977) Higher order monotonic convective difference schemes. *J. comp Phys.* **23,** 1-22.

Fox G., Johnson M., Lyzenga G., Otto S., Salmon J. and Walker D. (1988) *Solving Problems on Concurrent Processors Vol. I* Prentice-Hall, New Jersey.

Gazdag J. (1973) Numerical convective schemes based on accurate computation of space derivatives. *J. comp. Phys.* **13,** 100-113.

Gazdag J. and Canosa, J. (1974) Numerical solution of Fisher's equation. *J. Appl. Prob* **11,** 445-457.

Harley R. A., Russell A. G., McRae G. J., Cass G. R. and Seinfeld, J. H. (1993) Photochemical modeling of the Southern California air quality study. *Environ. Sci. Technol.* **27,** 378-388.

Hov Ø., Zlatev Z., Berkowicz R., Eliassen A. and Prahm L. P. (1989) Comparison of numerical techniques for use in air pollution models with nonlinear chemical

reactions. *Atmospheric Environment* **23,** 967-983.

Intel Corporation. (1991) A Touchstone DELTA system description. Intel Supercomputer Systems Division, Feb 26, 1991.

Lamb R. G. (1982) A regional scale model of photochemical air pollution, part 1: theoretical formulation. U. S. Environmental Protection Agency, Research Triangle Park, N. C.

Levin E. (1989) Grand Challenges to computational science. *Comm. ACM.* **32,** 1456-1457.

Lurmann F. W., Carter W. P. L., and Coyner L. A. (1987) A surrogate species chemical reaction mechanism for urban-scale air quality simulation models. EPA No. 68-02-4104.

McRae G. J., Goodin W. R. and Seinfeld J. H. (1982a) Numerical solution of the atmospheric diffusion equation for chemically reactive flows. *J. comp. Phys.* **45,** 1-42.

McRae G. J., Goodin W. R. and Seinfeld J. H. (1982b) Development of a second-generation mathematical model for urban air pollution. I. Model formulation. *Atmospheric Environment* **16,** 679-696.

Odman M. T., Kumar N. and Russell A. G. (1992) A comparison of fast chemical kinetic solvers for air-quality modeling. *Atmospheric Environment* **26,** 1783-1789.

Pai P. and Tsang T. T. H. (1992) Parallel computations of turbulent-diffusion in convective boundary-layers on shared-memory machines. *Atmospheric Environment* **26A,** 2425-2435.

Pilinis C. and Seinfeld J. H. (1988) Development and evaluation of an Eulerian photochemical gas-aerosol model. *Atmospheric Environment* **22,** 1985-2001.

Roache P. J. (1978) A pseudo-spectral FFT technique for non-periodic problems. *J. comp. Phys.* **27,** 204-220.

Rood R. B. (1987) Numerical advection algorithms and their role in atmospheric transport and chemistry models. *Rev. of Geophy.* **25,** 71-100.

Schere K. L. and Wayland P. K. (1991) EPA regional oxidant model (ROM 2.0): Evaluation on 1980 NEROS databases. U. S. Environmental Protection Agency,

Research Triangle Park, N. C.

Shin W. C., and Carmichael G. R. (1992) Comprehensive air pollution modeling on a multiprocessor system. *Comput. Ch. E.* **16,** 805-815.

Venkatram A., Karamchandani P. K. and Misra P. K. (1988) Testing a comprehensive acid deposition model. *Atmospheric Environment* **22,** 737-747.

Young T. R. and Boris J. P. (1977) A numerical technique for solving stiff ordinary differential equations associated with the chemical kinetics of reactive-flow problems. *J. Phys. Chem.* **81,** 2424-2427

Chapter 2

Numerical Advective Schemes Used in Air Quality Models—Sequential and Parallel Implementation

Donald Dabdub and John H. Seinfeld.

Atmospheric Environment, **28**, (1994) 3369–3385.

Six algorithms for solving the advection equation are compared to determine their suitability for use in photochemical air quality models. The algorithms tested are the Smolarkiewicz method, the Galerkin finite-element method, the numerical method of lines, the accurate space derivative method (ASD), Bott method, and Emde method. Four algorithms for filtering the numerical noise produced when solving the advection equation are also compared. The algorithms are evaluated both on two test problems and in the CIT model. The Galerkin finite-element and the ASD methods are implemented in the CIT in parallel computation. Results indicate that the ASD method, coupled with the Forester filter, produces the most accurate results. When the ASD transport solver is implemented in parallel, a speed-up of about 88 is achieved using 256 processors. Furthermore, a new set of optimized Forester filter parameters for grid-based air quality models is determined.

2.1 Introduction

Eulerian air quality models are based on the numerical solution of the atmospheric diffusion equation. Splitting methods provide an accurate and economical approach to solve the atmospheric diffusion equation (McRae *et al.,* 1982). The advection equation, one of the component operators in the splitting scheme, accounts for the transport of pollutants under a given wind field.

The two-dimensional advection equation is

$$\frac{\partial C}{\partial t} + \frac{\partial (uC)}{\partial x} + \frac{\partial (vC)}{\partial y} = 0, \tag{2.1}$$

where C is the concentration, t is time, and u, v are the x, y components of the wind velocity field. When solving Equation (2.1) numerically, it is well known that numerical diffusion and dispersion degrade the computed solution (Oran and Boris, 1987) as both the amplitude and phase of different Fourier components of the solution tend to be altered by numerical schemes. To overcome these errors, a large number of numerical advection schemes have been developed. Rood (1987) summarizes the development and improvements of many of the methods.

A variety of numerical advection schemes have been tested and compared to determine their suitability for use within air quality models (Chock and Dunker, 1983; Chock, 1985, 1991; Schere, 1983; Sheih and Ludwig, 1985; Tran and Mirabella 1991). Hov *et al.* (1989) show that, in AQMs, errors in the solution of the transport step are amplified during the chemistry step due to the highly nonlinear nature of the chemistry. Low order numerical schemes used to solve the advection equation provide non-oscillatory solutions with poor accuracy. High order numerical schemes, on the other hand, are characterized by computational noise near regions of steep gradients. The oscillatory noise increases in amplitude and propagates into neighboring grid points as the solution time increases, often producing negative values in the distribution being advected. Negative concentrations correspond to physically unrealistic "negative" mass. To overcome this problem different algorithms that "filter"

the oscillations have been proposed. A filter is a computational technique, used after each advection solver step, that removes computational noise. A filtered solution is expected to maintain the accuracy of high order numerical schemes, and present a distribution that is acceptable on physical terms.

This study has the following objectives: (1) To build on the work of Chock and others to test advection routines in both idealized tests and in a three-dimensional grid-based air quality model, the CIT model, (2) To evaluate nonlinear filters that have been proposed both on a test problem and in the three-dimensional CIT model, (3) To evaluate advection routines/filters in a parallel implementation of the CIT model, (4) To provide recommendations on the advection routines/filters of choice for three-dimensional air quality models, for both sequential and parallel environments.

2.2 Numerical Advection Algorithms

Equation (2.1) is frequently solved in air quality models using splitting methods (Yanenko, 1971), where (1) is approximated by the successive solution of

$$\frac{\partial C}{\partial t} + \frac{\partial (uC)}{\partial x} = 0 \tag{2.2}$$

$$\frac{\partial C}{\partial t} + \frac{\partial (vC)}{\partial y} = 0. \tag{2.3}$$

The character of Equation (2.1) and Equations (2.2) and (2.3) is that material is transported intact, without depletion or diffusion. Numerical methods solving Equations (2.1)-(2.3) must preserve this property as closely as possible.

A traditional way to evaluate the accuracy of numerical methods of solving Equations (2.2) and (2.3) is to advect an initial cosine hill distribution described by the following equations (Pepper and Long, 1978):

$$C_{i,j}(0) = \begin{cases} 50[1 + \cos \frac{\pi R}{4}] & \text{if } R < 4 \\ 0 & \text{for } R \geq 4 \end{cases} \tag{2.4}$$

$$R^2 = (x_i - x_0)^2 + (y_i - y_0)^2, \tag{2.5}$$

where $x_0 = 7, y_0 = 17$. The center of the hill is at grid point (7,17) with a concentration value of 100. Both x_i and y_i vary from 1 to 33. The hill rotates counter-clockwise about the $z-$axis at grid point (17,17) which is the center of the grid. The angular velocity is set so that it takes 7200π time units to complete one revolution. Time steps of 30π units were used in all the computations performed here.

Ideally after an integer number of revolutions the peak of the distribution should be 100 and its center should be located at (7,17). To evaluate the accuracy of different schemes, the following measures have been traditionally used (Chock and Dunker 1983; Chock 1985, 1991; Tran and Mirabella 1991):

$$\text{Mass Conservation Ratio} = \sum_{i,j} C_{i,j}(t) / \sum_{i,j} C_{i,j}(0) \tag{2.6}$$

$$\text{Mass Distribution Ratio} = \sum_{i,j} C^2_{i,j}(t) / \sum_{i,j} C^2_{i,j}(0) \tag{2.7}$$

$$\text{Maximum Absolute Error} = \max(|C_{i,j}(t) - C^e_{i,j}(t)|) \tag{2.8}$$

where $C_{i,j}$ represents the computed concentration at grid point i, j, and $C^e_{i,j}$ is the exact concentration at grid point i, j. The mass conservation ratio and the mass distribution ratio measure the amount of numerical diffusion. Note that a method might present a good mass conservation ratio while maintaining a poor mass distribution. The average absolute error measures an average discrepancy of the mass field from the exact solution. Finally, the maximum absolute error is most sensitive to the displacement and height of the distribution's peak.

Table 2.1 lists the methods examined in this study. SMOL and GLRK were selected since they are currently implemented in air quality models—the Urban Airshed Model (UAM) (Morris and Myers, 1990) and the CIT model (Harley *et al.*, 1993), respectively. The numerical method of lines (NMOL) is selected since it has not been

compared previously with other methods in a systematic manner and has several potentially desirable attributes. For instance, NMOL converts the PDE into a system of ODEs in a simple systematic manner. NMOL can then take advantage of recent progress in the numerical solution of ODEs. The accurate space derivative method (ASD) is selected as an alternative method to those currently used in existing AQMs that provides greater accuracy (Chock, 1991). Other alternative methods included in this study are the Bott solver and then Emde solver. The use of parallel computers opens up the practical uses of the ASD method, since it requires significantly greater computational time on a sequential machine than the other methods in Table 2.1. Some of the methods have been previously compared by Chock and Dunker (1983) and Chock (1985, 1991) The present study builds on Chock's results and evaluates several of the methods in a full three-dimensional air quality model, under both sequential and parallel implementation.

Most of the methods listed in Table 2.1 are well described in the literature. Therefore, only the particular implementation of NMOL used in this study will be discussed. The numerical method of lines approximates the spatial derivatives of a partial differential equation with an appropriate finite-difference algebraic expression (Schiesser, 1991). The time derivative of the PDE is left unmodified, leading to a system of ordinary differential equations in time. The NMOL implementation used in this study consists of a fourth-order directional finite-difference approximation to discretize the spatial derivatives. Namely, for positive wind velocity

$$\frac{\partial C_i}{\partial x} = \frac{-C_{i-3} + 6C_{i-2} - 18C_{i-1} + 10C_i + 3C_{i+1}}{12\Delta x}. \tag{2.9}$$

Similarly, for negative wind velocity

$$\frac{\partial C_i}{\partial x} = \frac{-3C_{i-1} - 10C_i + 18C_{i+1} - 6C_{i+2} + C_{i+3}}{12\Delta x}, \tag{2.10}$$

where C_i represents the concentration at grid point i in the one-dimensional sense when solving Equations (2.2) and (2.3). It is well known that Equations (2.9) and

Table 2.1: Numerical advection methods considered and concentrations and locations of the peak of a cosine hill after two revolutions.

Method	**Characteristics**	**Reference**	**Peak**[†]
SMOL Smolarkiewicz Method	Iterative scheme based on upstream differences. Positive definite.	Smolarkiewicz (1983)	26 @ (8,19)
NMOL Numerical Method of lines.	Fourth-order directional difference in space. SDRIV2 integration in time. Produces negative conc.	Schiesser (1991) Carver and Hinds (1978) Kahaner *et al* (1991)	59 @ (7,16)
FNMOL Numerical Method of lines filtered	NMOL with Forester filter. Filter parameters: $K = 1$, $\mu = 0.1$, $m = 1$, and $n = 2$		44 @ (7,17)
BOTT Bott Method	Nonlinear renormalization of advective fluxes. Positive definite	Bott (1989)	73 (7,17)
EMDE Emde Method	Continuous curvature cubic-spline. Positive definite.	Emde (1992)	80 (7,17)
GLRK Galerkin Method	Chapeau function Galerkin finite element. Produces negative conc.	MacRae *et al* (1982)	90 @ (7,18)
FGLRK Galerkin Method filtered	GLRK with Forester filter. Filter parameters: $K = 1$, $\mu = 0.1$, $m = 1$, and $n = 2$		75 @ (7,17)
FGLRK2 Galerkin Method filtered	GLRK with Forester filter. Filter parameters: $K = 3$, $\mu = 0.2$, $m = 2$, and $n = 4$		43 @ (7,17)
ASD Accurate Space Derivative Method	Fourier techniques to accurately compute space derivatives. Third-order Taylor expansion to integrate in time. Produces negative conc.	Gazdag, (1973)	98 @ (7,17)
FASD Accurate Space Derivative Method filtered	ASD with Forester filter. Filter parameters: $K = 1$, $\mu = 0.1$, $m = 1$, and $n = 2$		98 @ (7,17)

[†]Exact value: 100 @ (7,17)

(2.10) are superior to central finite-differences from the accuracy point of view (Carver and Hinds, 1978). The resulting system of ordinary differential equations is solved using the SDRIV2 integrator described by Kahaner *et al.* (1991).

2.3 Nonlinear Filters Used in the Solution of the Advection Equation

As mentioned earlier, the numerical solution of the advection equation presents several difficulties. High-order algorithms produce spurious waves near sharp gradient regions that produce physically unrealistic negative concentrations. To design a filter, one is faced with the question: What is to be done with the "negative" mass that appears in the numerical solution? The simple idea of setting the negative values to zero is not the proper approach, because it is not mass conservative; rather "negative" mass must be redistributed over positive concentrations in the solution. However, there is no specific mathematical or physical guideline on how to perform such a redistribution. To address this problem, various "filtering" techniques have been proposed to remove the negative concentrations and to smooth the positive oscillatory intervals of the distribution.

There are two main approaches to achieve such goals. The first approach consists of selectively introducing nonlinear diffusion to the distribution. The second approach consists of scanning local maxima and local minima on the distribution and adjusting selected distribution values iteratively. The filter step is an important part of the advection computations since the numerical results can be strongly affected by the presence of a filter. Various available filters that can be used in the advection routines in AQMs have not been compared previously. This section compares different filtering techniques that have been proposed in the literature with the goal of determining an optimal filter for use in AQMs.

Let $\mathcal{A}$ be an algorithm (finite-difference, finite element, spectral, etc.) to solve

Equation (2.2), expressed in the form,

$$C_i^{t+\Delta t} = \mathcal{A}(\mathbf{C}^t, u), \tag{2.11}$$

where the subscript i refers to grid block i. The redistribution process consists of coupling $\mathcal{A}$ with a filter $\mathcal{F}$ in the following way:

$$B_i^t = C_i^t$$

$$B_i^{t+\Delta t} = \mathcal{A}(\mathbf{B}^t, u) \tag{2.12}$$

$$C_i^{t+\Delta t} = \mathcal{F}(\mathbf{B}^{t+\Delta t}, C_i^t).$$

$\mathcal{F}$ must satisfy several conditions that arise naturally in air pollution models:

(1) *Mass conservation.* The filter should not add or remove mass from the system,

$$\sum_i C_i^{t+\Delta t} = \sum_i C_i^t. \tag{2.13}$$

(2) *Positiveness.* The filter should remove all negative values completely,

$$C_i^{t+\Delta t} \geq 0 \text{ for all } i. \tag{2.14}$$

(3) *Total Variation Diminishing.* The filter should guarantee that there are no spurious oscillations near the regions of sharp gradients. The criterion commonly used is the TVD inequality:

$$TV(C^{t+\Delta t}) \leq TV(C^t) \tag{2.15}$$

where

$$TV(C^t) = \sum_i |(C_{i+1}^t - C_i^t)|.$$

(4) *Shape Preservation.* The filter should modify the minimum number of grid points to enforce conditions (1) through (3).

Table 2.2: Filters for numerical advection methods.

Filter	Characteristics	Reference
Bartinicki	Scans negative distribution values. Guarantees absence of negative values. Not TVD	Bartinicki (1989)
Chapman	Introduces local diffusion. Mass conservative	Chapman (1981)
Engquist	Scans local extrema. TVD Mass conservative	Engquist *et al.* (1988)
Forester	Introduces local diffusion. Mass conservative. Requires four problem dependent filter parameters.	Forester (1977)

An appropriate test case used to study the performance of different filters is a simple one-dimensional advection problem with constant wind velocity. Three different initial conditions, square, triangle, and cosine hill were selected here. For each initial condition Courant numbers of 1, 0.5, 0.1, and 0.01 were used. The Courant number is defined as $(u\Delta t)/\Delta x$.

Table 2.2 summarizes the filters studied. A more detailed description of the filters is given in the Appendix.

2.4 Evaluation of the Performance of Advection Algorithms and Filters on Test Problems

This section evaluates the different advection schemes and nonlinear filters described previously. The evaluation of advection schemes on test problems is kept brief here as this aspect has been presented in detail by Chock and Dunker (1983) and Chock (1985, 1991). As noted above, the evaluation of nonlinear filters for AQMs, on the other hand, has not been studied previously. Finally, the selection of optimal filter parameters is discussed.

2.4.1 Advection Schemes

When solving the advection equation some algorithms exhibit non-physical oscillations and/or negative concentrations near steep gradient regions of the solution. To overcome this difficulty the comparison of advection schemes that we will present will utilize the Forester filtering (Forester, 1977) with filter parameters, $K = 1$, $\mu = 0.1$, $m = 1$, and $n = 2$. As we will show shortly, this set of Forester filter parameters achieves the best overall accuracy. K is the number of filtering iterations, and μ is a dimensionless diffusion coefficient. The value of m represents half the wavelength of the lowest frequency noise. The value of n does not represent any physical aspect of the filter. It is simply a numerical parameter that must be large enough to permit continuity of nonzero C values. The emphasis of the comparison in this section is on the advection solver performance. The choice of the Forester filter or a particular set of filter parameters does not give an undue advantage to any one method. If another filter is selected, the relative accuracy of the advection algorithms will remain unmodified.

Figures 2.1 and 2.2 show the mass conservation ratio as a function of number of revolutions of the test cosine hill for the different advection solvers studied. The Forester filter slightly improves the mass conservation properties of the schemes; without Forester filtering, most of the schemes still present an acceptable performance from the point of view of conserving mass. However, the SMOL method, after one revolution, is quite mass dissipative. At the other extreme, the NMOL solver is slightly biased to a higher mass conservation ratio. Figures 2.1 and 2.2 do show undesirable properties of SMOL. However, the mass conservation ratio does not present sufficient information to discern the methods' relative accuracies.

Figures 2.3 and 2.4 show the mass distribution ratio as a function of number of revolutions of the test cosine hill. A low mass distribution ratio indicates a high amount of artificial diffusion present in the numerical scheme. It can be observed that the use of Forester smoothing tends to decrease the mass distribution ratio in all the schemes. ASD seems to be the scheme least affected when filtering is added.

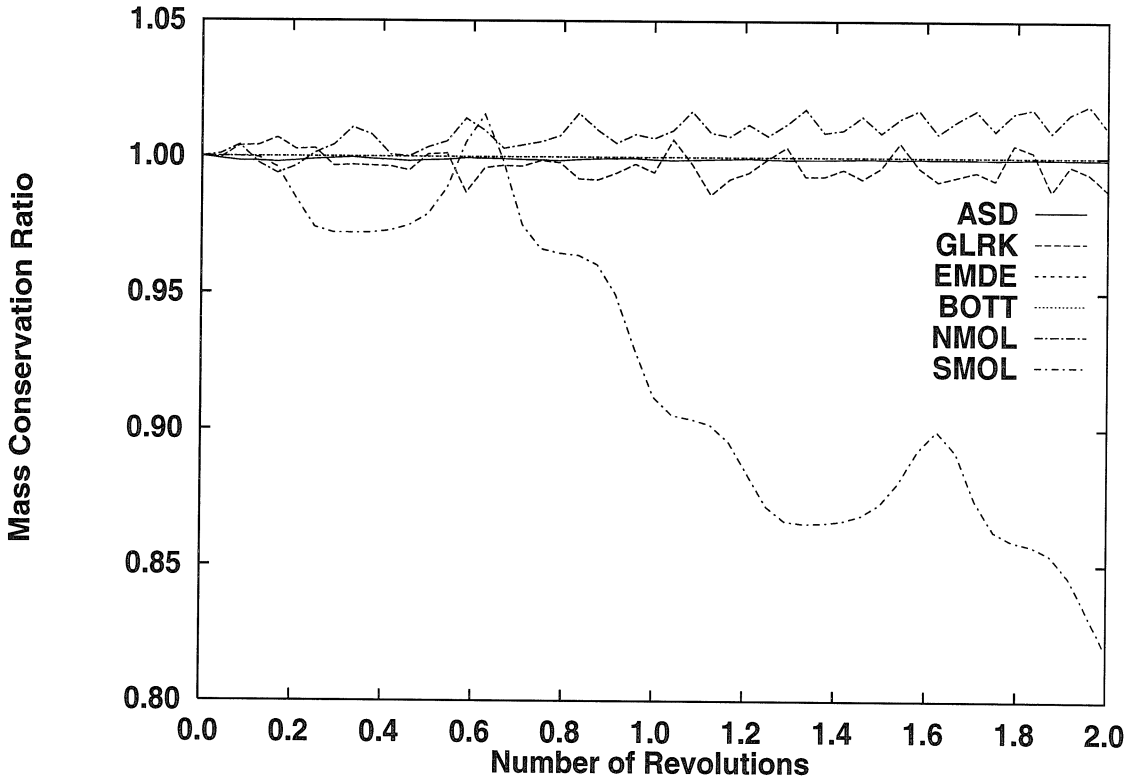


Figure 2.1: Mass conservation ratio for unfiltered advection solvers for the rotating cosine hill problem using a time step of 30 π.

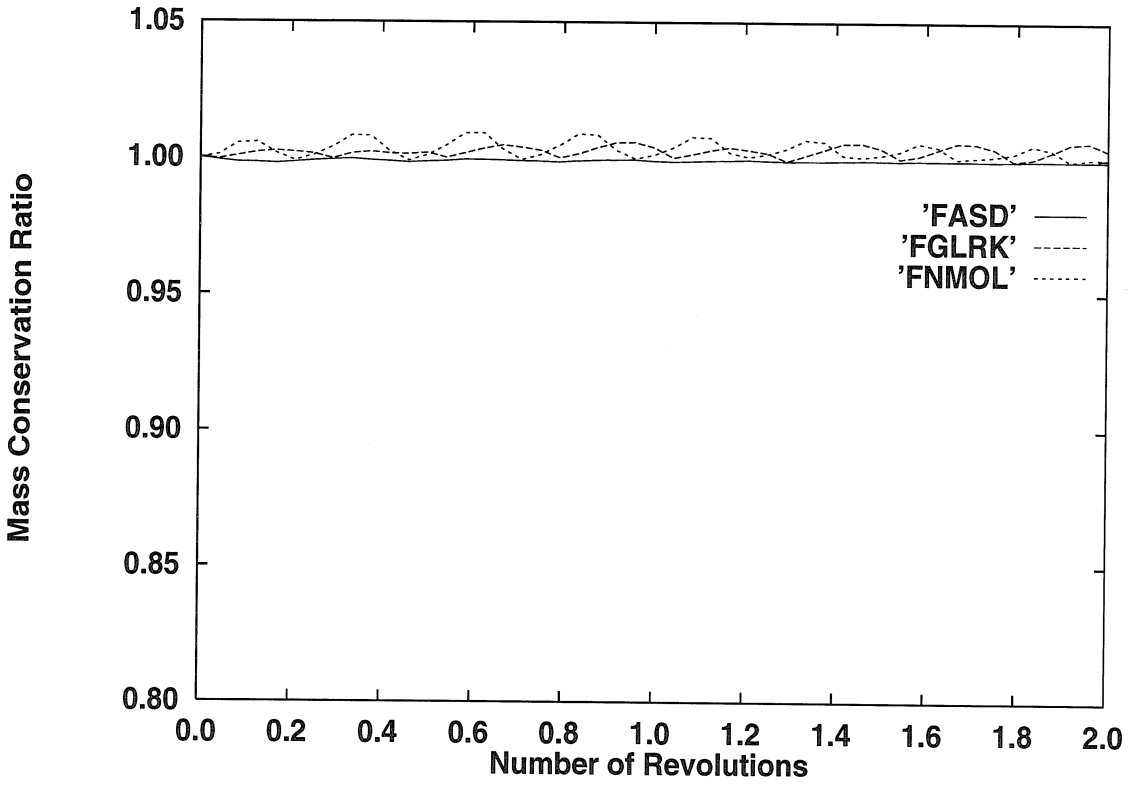


Figure 2.2: Mass conservation ratio for Forester filtered advection solvers for the rotating cosine hill problem using a time step of 30 π.

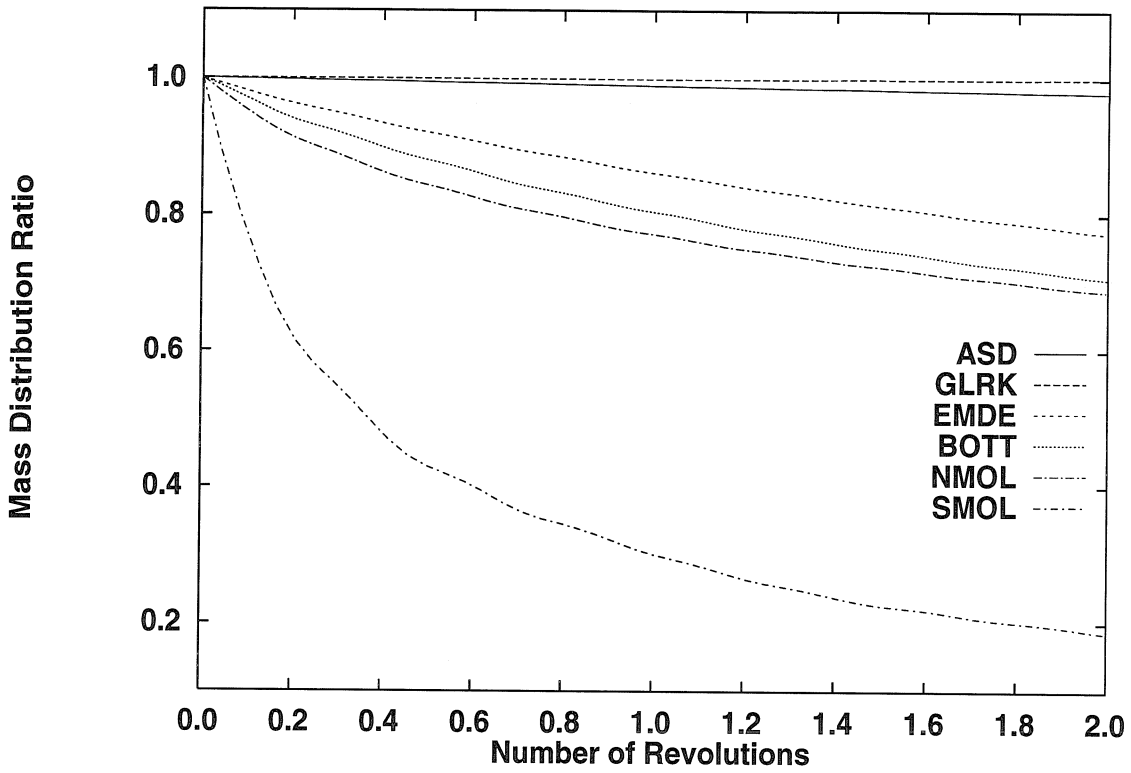


Figure 2.3: Mass distribution ratio for unfiltered advection solvers for the rotating cosine hill problem using a time step of 30 π.

SMOL presents the poorest mass distribution ratio of all the schemes studied. Furthermore, Figure 2.4 shows that regardless of its acceptable mass conservation ratio, the NMOL scheme has a lower mass distribution ratio than the ASD or Galerkin schemes. Furthermore, it shows that regardless of their almost perfect mass conservation ration, EMDE and BOTT solvers have a mass distribution ration worse than ASD and GLRK methods. The mass distribution ratio of the solvers parallels their preservation of peak values as reported in Table 2.1.

Table 2.1 presents the resulting distribution of peak magnitude and location after two revolutions for all the schemes studied. The ASD method presents the best peak preservation properties. The computation time used by the GLRK filtered is comparable to that of the SMOL algorithm. The computation time used by NMOL filtered and ASD filtered is about an order of magnitude greater. Results presented

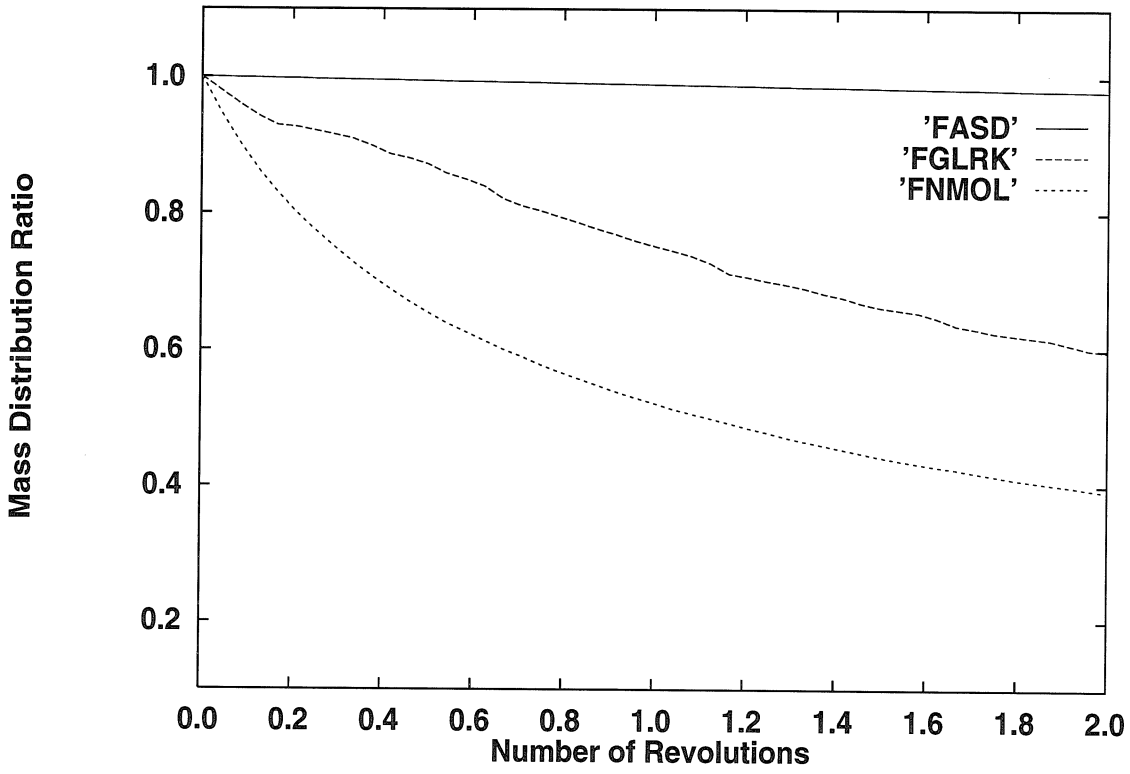


Figure 2.4: Mass distribution ratio for Forester filtered advection solvers for the rotating cosine hill problem using a time step of 30 π.

in Table 2.1 agree with those presented by Chock (1983) and Chock (1985, 1991).

A more challenging test problem, as suggested by Odman and Russell (1993), was performed. It consists of using the following velocity field:

$$u_r = 0$$

$$u_\theta = \omega R[1 - (R/R_{max})^2] \tag{2.16}$$

where u_r and u_θ represent the radial and angular components of the wind field; ω is the angular velocity at the peak, adjusted so that the peak completes one full rotation in 240 time steps; R_{max} is the maximum of R as defined in Equation (2.5). This advective field is characterized by a parabolic angular velocity profile. A parabolic profile is more challenging than the rigid body rotation counter-clockwise rotation since it does not yield constant velocity components along straight lines. Results obtained using the parabolic angular velocity profile present the same relative accuracy for all solvers as that described in Table 2.1. However, all the solvers present a decrease in accuracy as expected.

2.4.2 Nonlinear Filters

Figures 2.5 and 2.6 show the performance of the different filters for Courant numbers of 1 and 0.1, respectively as a function of number of integration steps. The error presented equals that of the unfiltered solution. The error norms are defined as

$$|\text{error}|_{L_1} = \sum_i (C_i(t) - C_i^e(t)) \tag{2.17}$$

$$|\text{error}|_{L_\infty} = \max(|C_i(t) - C_i^e(t)|). \tag{2.18}$$

The error in the L_1 norm is an indication of the average accuracy of the solution. The error in the L_∞ norm is an indication of the peak conservation of the solution for the initial conditions used in this problem. From the computational point of view, the

Courant number determines the upper bound of the number of neighboring points that contain information needed to advance the local solution. Results presented in Figures 2.5 and 2.6 correspond to the Galerkin algorithm using a square wave pulse as initial condition. The pulse width is 20 grid points. For reasons of space, and since other initial conditions and other advection solvers yield similar results, Galerkin results are the only ones reported. It was observed that all the filters except for Bartnicki maintain an excellent mass conservation ratio for all Courant numbers studied. Forester filters present the lowest error in the l_1 and l_∞ norm sense for a Courant number of 1 as shown in Figures 2.5a and 2.5b. Note that for Courant number of 1 the Chapman filter has no effect on the distribution. For a Courant number of 0.1, the Forester filter can still be considered the best filter to be used based on its error as shown in Figures 2.6a and 2.6b. However, it was observed that for a Courant number of 0.01 the Chapman filter performs somewhat better than the rest of the filters. This Courant number is not a typical condition found in AQMs.

One of the disadvantages of the Forester filter is that it requires four parameters that are dependent on the problem to be solved. These parameters can have a strong effect on the accuracy of the advection scheme being filtered as shown in Table 2.1 in the FGLRK and FGLRK2 entries. It is observed that using different values of such parameters can reduce the peak height by more than 40%. Therefore, filter parameters should be carefully selected for a given problem. Forester (1977) attempted to determine by analysis the proper parameter values for the adequate control of computational noise. However, the problem becomes too complex and he concluded that empirical tests should be used. The parameters must be chosen such that each is as small as possible without producing negative concentration values. The values of K and μ should be small so that artificial diffusion introduced by the filter is minimized. The values of m and n should be the smallest values that permit continuity of nonzero values of the distribution. We have performed an extensive trial-and-error evaluation to determine the optimal set of Forester filter parameters using this test problem. The optimal set of parameters is: $K = 1$, $\mu = 0.1$, $m = 1$, and $n = 2$.

Each of the filters described can be applied to any advection solver. Indeed, the

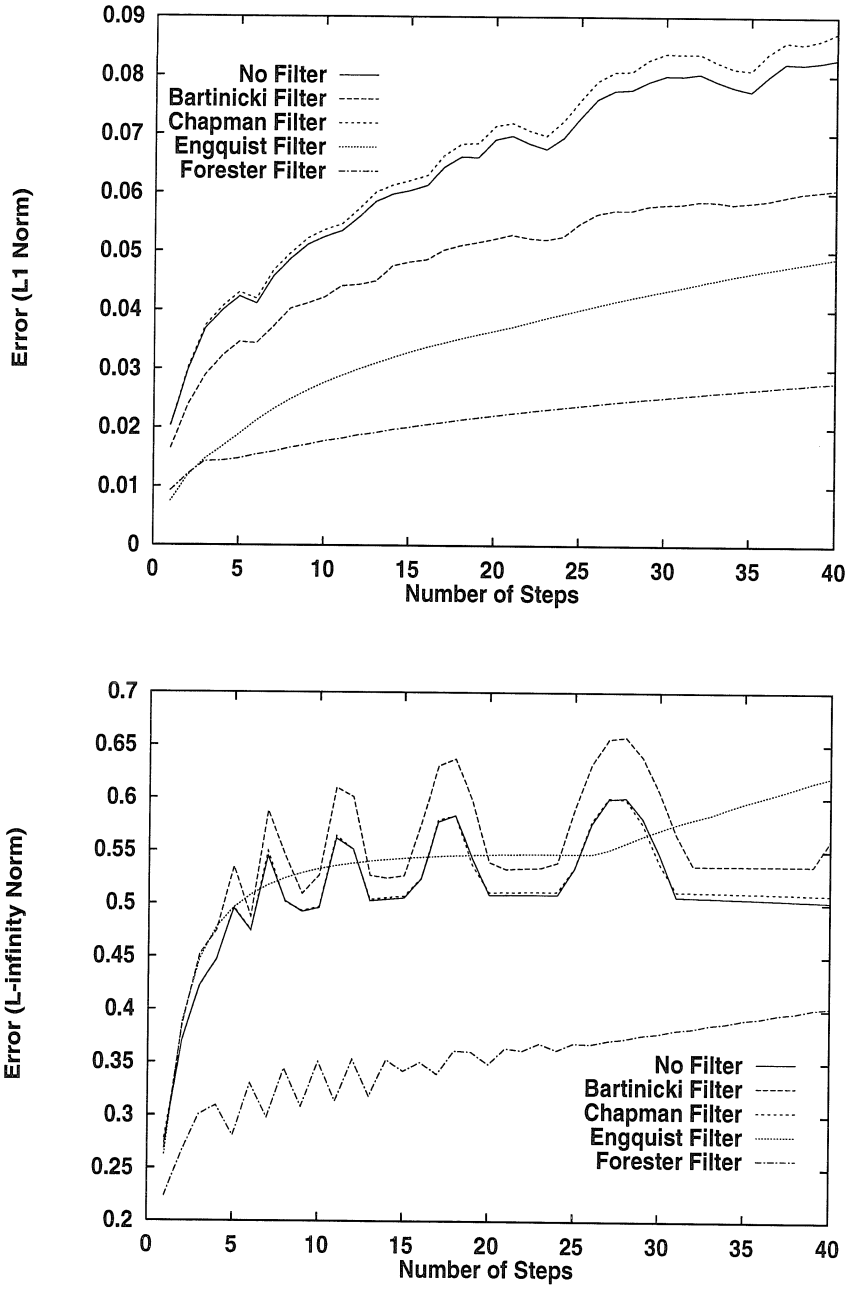


Figure 2.5: (a) L_1 error norm for the solution to the one-dimensional advection equation with square pulse initial conditions using a Galerkin solver and various filters. (Courant number 1.0) (b) L_∞ error norm for the solution to the one-dimensional advection equation with square pulse initial conditions using a Galerkin solver and various filters. (Courant number 1.0)

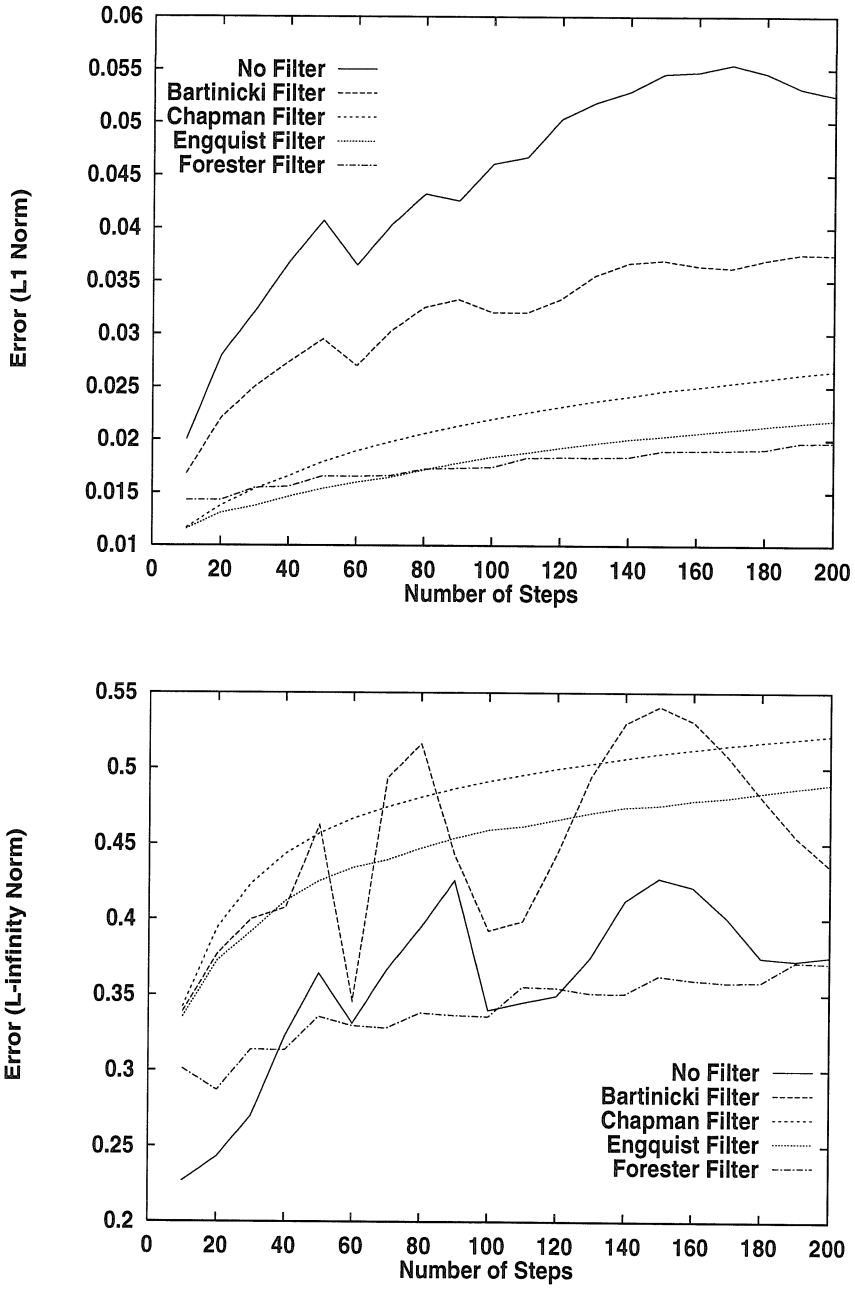


Figure 2.6: (a) L_1 error norm for the solution to the one-dimensional advection equation with square pulse initial conditions using a Galerkin solver and various filters. (Courant number 0.1) (b) L_∞ error norm for the solution to the one-dimensional advection equation with square pulse initial conditions using a Galerkin solver and various filters. (Courant number 0.1)

advection solver and the filter are treated as two independent modular computational routines in this study. Nevertheless, Odman and Russell (1993) present a taylor-made filter for multi-dimensional finite element methods. It has been observed in our test runs that a given filter has the same relative effects on all the advection solvers evaluated. In summary, the combination of ASD with the Forester filter shows the best performance for most of the cases studied. At a low Courant number (0.01), which is not as relevant for air quality modeling applications as the other values of the Courant number studied here, the Chapman filter combined with ASD presents the best overall performance.

2.5 Evaluation of the Performance of Advection Algorithms and Filters in the CIT Model

As shown in Table 2.1 and Figures 2.1-2.4, the test problems indicate that ASD is the most accurate of the methods tested. This section discusses the implementation of the ASD method in the CIT model and comparison of the ASD and GLRK methods. This comparison shows the impact of the advection solver, interacting with chemistry, on peak pollutant concentrations. On the basis of the cosine hill tests, SMOL and NMOL methods have been eliminated from further consideration.

Initial test problems also show that the use of somewhat different Forester filter parameters with the Galerkin solver has dramatic effects on the accuracy of the solution. The ozone concentration predictions of the CIT model with the Galerkin solver using two sets of Forester filters parameters will also be compared. One set corresponds to the filter parameters implemented in the CIT model while the other set corresponds to optimized parameters determined on the test examples. This comparison shows the impact of the filter parameters when transport and chemistry interact within an AQM. As a result of the comparison, a new set of Forester filter parameters is recommended to be used with the CIT model.

2.5.1 Implementing the ASD Method in a Three-Dimensional AQM

The horizontal boundary conditions used to solve the advection equations within an air quality model are

$$(uC) \cdot \hat{n} = (u_b(t)C_b(t)) \cdot \hat{n} \qquad u \cdot \hat{n} \leq 0 \tag{2.19}$$

$$-\nabla C \cdot \hat{n} = 0 \qquad u \cdot \hat{n} > 0 \tag{2.20}$$

where $\hat{n}$ is the normal to the boundary, and $u_b(t)$ and $C_b(t)$ are the specified wind speed and concentration at the boundary. If a function is smooth and periodic its Fourier series does not exhibit the Gibbs phenomenon. Since ASD involves Fourier transforms, it requires periodicity to avoid such Gibbs phenomenon. Therefore, to implement ASD into a three-dimensional AQM, first one must use some computational "artifices" while performing FFTs to meet the periodicity requirement.

There have been different approaches developed to meet the need for periodicity. Roach (1978) describes a technique termed "reduction to periodicity." It consists of splitting the value of the concentrations, $C(x)$ into a periodic function, $F(x)$ and a polynomial of a given degree, $P(x)$, $C(x) = P(x) + F(x)$. The coefficients of the polynomial $P(x)$ are chosen so that the residual $F(x)$ has periodic derivatives at the boundaries. To compute the spatial derivatives of $C(x)$ the fast Fourier transform (FFT) is applied to $F(x)$ only. The derivatives of $P(x)$ are obtained analytically. Chock (1991) describes an alternative technique called "periodicity recovery." It consists of extending the domain of the solution and using a polynomial or spline function fitted to assure periodicity. Wengle and Seinfeld (1978) proposed to expand $C(x)$ into Chebyshev polynomials. Finally, Gazdag (1973) used mirror techniques that consist of doubling the domain with the mirror image of the data to be transformed. The ASD transport solver was implemented into the CIT photochemical model using periodicity recovery.

Currently, the chemistry solver for AQMs is the most computationally intensive

part of the numerical solution. For instance, the CIT model with the Galerkin advection routine spends about 90% of its CPU time on the chemistry solver and approximately 5% on the horizontal transport solver. Implementing ASD into the CIT model, as executed on a sequential computer, causes the full model to run four times slower than when using the Galerkin advection solver. The reason that ASD requires greater CPU time than Galerkin is that ASD performs 3 FFTs and 3 inverse FFTs computations per time step. On the other hand, the Galerkin method only solves a tridiagonal system per time step. Table 2.3 presents detailed timing data for a 24-hour run of the CIT model on an IBM RISC 580 (sequential architecture) for different modeling cases.

2.5.2 Evaluation of the Galerkin and ASD Methods in the CIT Model—Sequential Implementation

The simulations reported here are for the same conditions as those reported in Harley *et al.* (1993) for August 27, 1987 in the South Coast Air Basin. The reader is referred to Harley *et al.* (1993) for all details of the simulation.

Figure 2.7 shows the predicted ground-level ozone concentrations in the South Coast Air Basin at 14:00 hours on August 27, 1987. The computations were performed using the Galerkin solver and Forester filter with the parameters originally used in the model, $K = 3$, $\mu = 0.2$, $m = 2$, and $n = 4$. Previous tests (FGLRK and FGLRK2 entries in Table 2.1) indicate that these parameters introduce excessive artificial diffusion in by the filtering step. Figure 2.8 shows the predicted ground-level ozone concentrations for the same conditions as in Figure 2.7 with the only change being that the filter parameters are selected as $K = 1$, $\mu = 0.1$, $m = 1$, and $n = 2$, those found in the test example described above to produce the best concentration peak results. It is observed that with the new filter parameters an ozone peak appears in the north west region of the modeled area, the ozone peak in the eastern part of the region expands in size, and there is a small decrease in the ozone peak of the south central region of the domain. Figure 2.9 shows CIT

Table 2.3: Performance and CPU usage distribution for a 24-hour simulation of the South Coast Air Basin under different numerical schemes.

Transport	Chemistry	Total Time (s)	Chemistry Integration (s)	Other Comput. (s)	Relative Speed	Chemistry Comput. (s)%	Other Comput. %
Galerkin	On	4,419.08	3,967.66	451.42	10.52	89.78	10.22
Galerkin	Off	419.94	0.00	419.94	1.00	0.00	100.00
ASD	On	17,507.12	3,975.30	13,531.82	41.69	22.71	77.29
ASD	Off	13,327.46	0.00	13,327.46	31.74	0.00	100.00

model predictions using the ASD method as the advection solver with the new filter parameters. Ozone maxima are predicted in the same location as with the Galerkin method with the corrected filter parameters. However, when the ASD method is used the ozone maxima in the eastern portion of the South Coast Air Basin are enhanced by about 20 ppb, attributable entirely to the better properties of the ASD method relative to the Galerkin method. These results indicate that the choice of advection routine in a photochemical air quality model can have a measurable effect of predicted levels of ozone and other species.

2.6 Parallel Implementation of Advection Schemes

The use of the ASD method requires a significantly greater amount of computer time than the Galerkin or Smolarkiewicz methods currently implemented in AQMs. This is a major practical disadvantage if sequential computers are to be used. To overcome this problem ASD can be implemented on a parallel computer. The basic idea behind the parallel implementation of the advection solver is to perform the transport computations of all rows or columns simultaneously. This section discusses three approaches to implement the transport computations in parallel: Extended Arrays (EA), Designated Transport Nodes (DTN) and Dynamically Balanced (DYB).

The solution of the advection equation in parallel is more challenging than the integration of the chemistry portion of an AQM. Transport calculations inherently require non-local data (i.e., concentration values, wind, etc.) to be able to predict the grid concentration values after each transport step. Furthermore, AQMs typically spend less than 10% of their CPU time solving the advection operator. Due to the challenges presented by parallelizing transport and to the relatively small computational time required by most advection solvers, one might conclude at first thought that perhaps it is not effective to perform transport computations in parallel. This is not the case. The speed-up predicted by Amdahl's Law becomes increasingly sensitive to the fraction of the code that is already parallelized. In the case of air quality models, in which about 90% of the computations are parallelized, the speed-up gained by

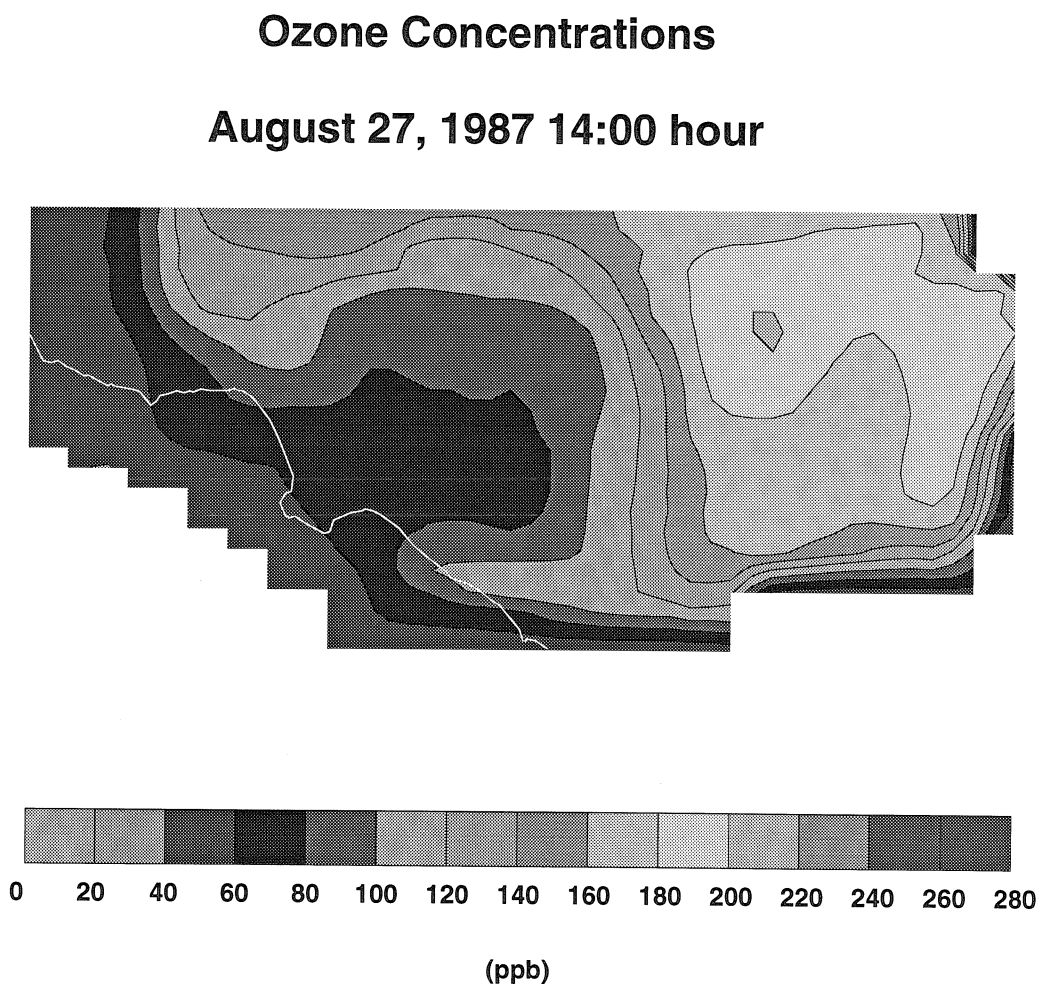


Figure 2.7: Ozone concentrations at 14:00 hour (August 27,1987) predicted by the CIT model using the Galerkin advection solver and filter parameters: $K = 3$, $\mu = 0.2$, $m = 2$, and $n = 4$

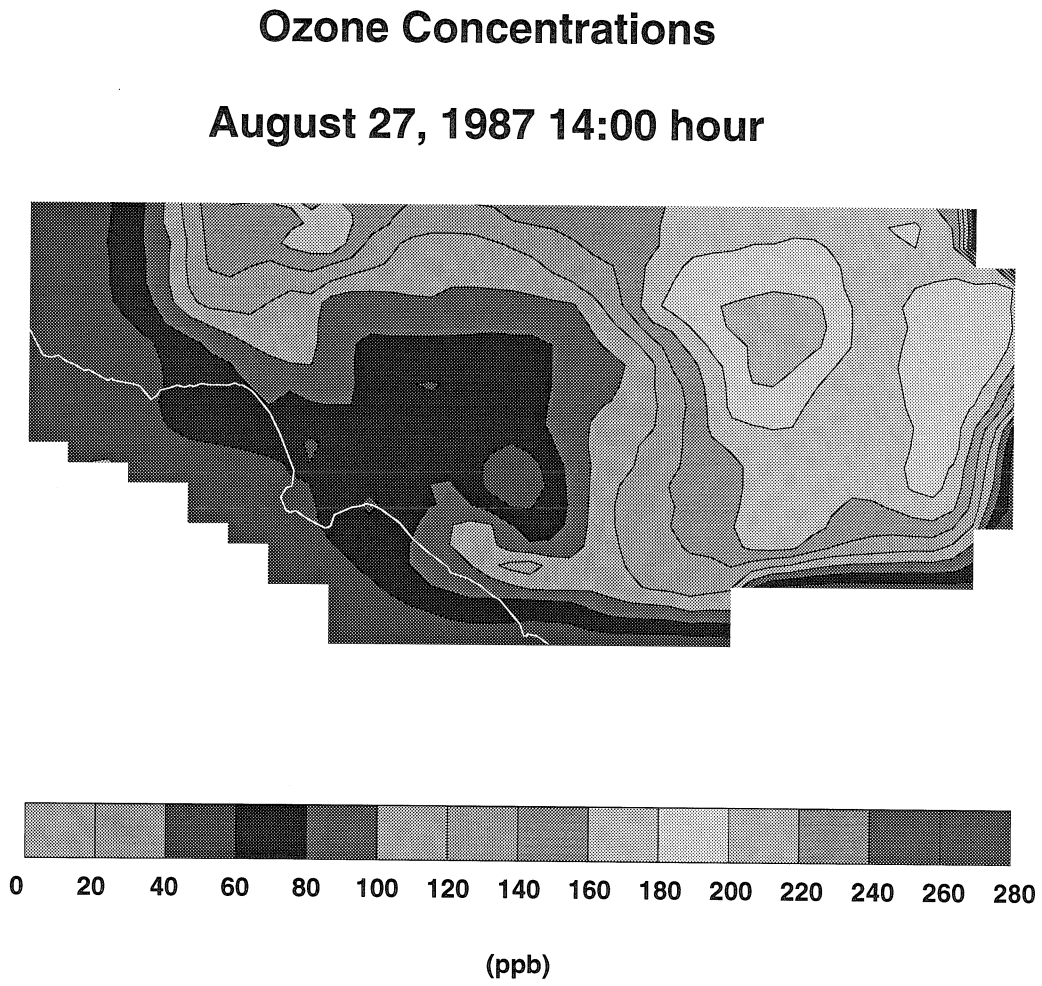


Figure 2.8: Ozone concentrations at 14:00 hour (August 27,1987) predicted by the CIT model using the Galerkin advection solver and filter parameters: $K = 1$, $\mu = 0.1$, $m = 1$, and $n = 2$.

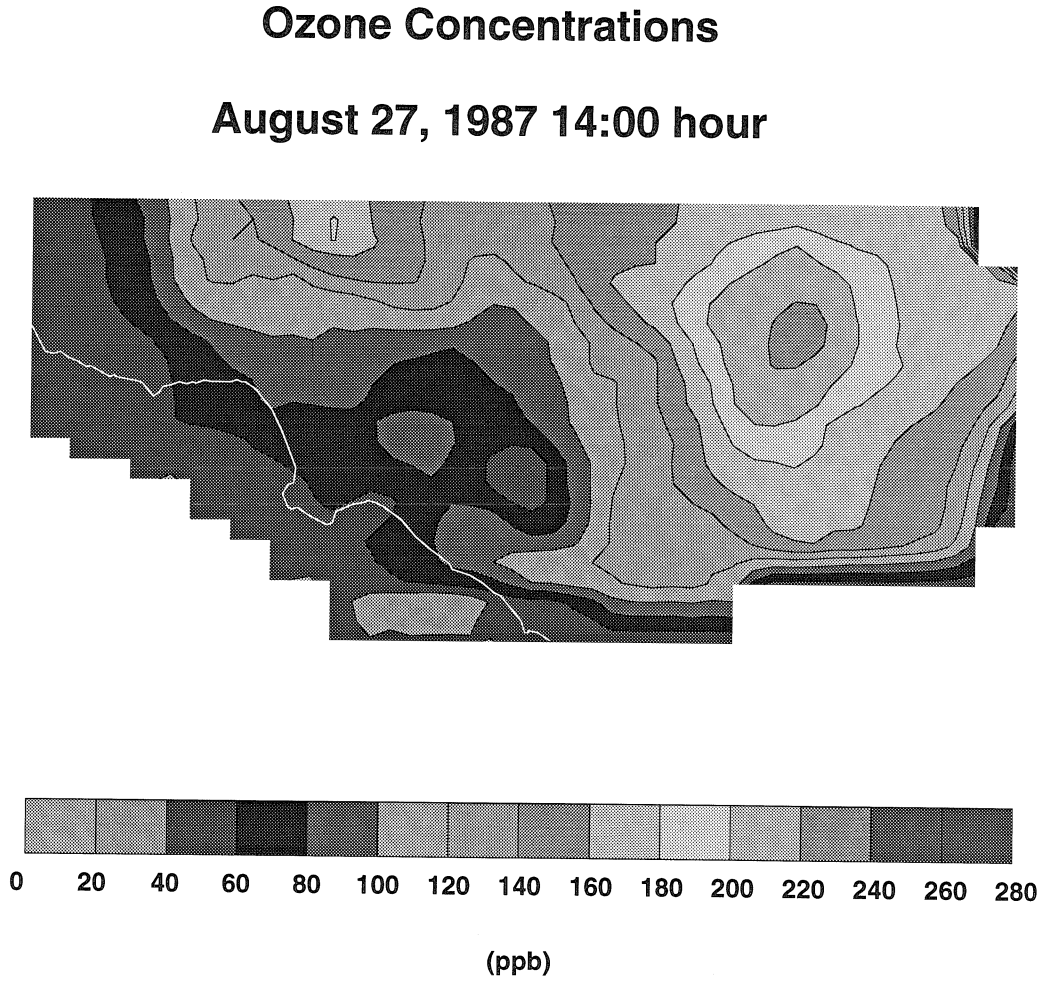


Figure 2.9: Ozone concentrations at 14:00 hour (August 27,1987) predicted by the CIT model using the ASD advection solver and filter parameters: $K = 1$, $\mu = 0.1$, $m = 1$, and $n = 2$.

implementing transport computations in parallel is predicted to become significant.

The numerical solution of the advection equation requires neighbor grid cell data that are not always available in local nodes. The number of neighboring grid cells required is dependent on the underlying transport algorithm used. For instance, a second-order finite element method needs two neighboring grid cells in each direction to predict the local grid cell concentration. A spectrally accurate method, like ASD, needs all the data of the entire row or column to be able to predict any local grid cell concentration. The reason for such data dependency is that spectral methods involve global operations like the transformation of data to the Fourier domain. Finite element algorithms and spectral methods are perfect examples to show the wide differences in data flow structures imposed by two transport solvers. The optimum technique to implement is dependent on the numerical method used to carry the transport step, the degree of modularity and portability desired, and the number of nodes available.

The following are desirable characteristics for the ideal implementation of transport solvers in a parallel environment:

(1) *Minimum Communication.* The message passing time spent on interprocessor communications is expensive relative to computational time. It is desirable to pass a single long message rather than an equivalent message size using various send/receive operations, the reason being that every time a message is passed there is initialization overhead involved in the process. The ideal implementation would minimize the data flow among nodes.

(2) *Portability.* The parallel implementation should contain only simple synchronous or asynchronous sends and receives. It should avoid global message operations such as broadcast, global sums, etc. The reason for such constraints is that all the message passing protocols (e.g. PVM, P4, NX, EXPRESS) support such operations allowing the code to be easily ported among different parallel compilers.

(3) *Modularity.* The code should be modular in the sense that the transport solver of the model is a single routine that can be easily replaced by a different algorithm. In addition, the parallel code should be written in such a way that the data flow

between nodes, for any number of nodes, remains unmodified for any advection solver. Modularity should be maintained while keeping the internode communication to a minimum as described in (1).

(4) *Load Balance.* To perform a horizontal transport step it is necessary for a particular node to receive non-local data from one or more other neighboring nodes. At times, some of the neighbor nodes are still performing other computations and cannot send the data at the exact time of request. As a result, the node requesting the data stays idle until the neighbor node is able to perform the send operation. The ideal parallel implementation would minimize such idle time spent between sends and receives of data among all nodes.

Extended Arrays, EA (Fox *et al.,* 1988), is an implementation that minimizes internode communication while optimally maintaining the CPU load balance. The name implies that local concentration arrays are extended at the boundaries to make room for the data needed. Each node sends the boundary concentration data to all its neighbors. The number of grid cells sent as boundaries is the minimum one imposed by the advection solver used. Each node will, of course, receive data from its neighbors. After the data are received all the nodes perform the transport step locally in small domains. The use of EA in AQMs is recommended when the algorithm used requires a few grid cells located in neighbor nodes and speed is the primary concern. The EA approach to implementing the transport is not modular. When using EA techniques, changing the algorithm requires extensive modifications to the parallel implementation.

To overcome modularity problems presented by the EA approach one can designate a particular node that receives not only the boundary of grid cells located in the neighbor nodes, but all the grid cells of a particular row or column. After receiving the data, the designated transport node performs the transport computations and then sends back the new grid cell concentrations to the appropriate nodes. All columns or rows are being processed simultaneously at a given horizontal transport step. This approach is named Designated Transport Node, DTN. The results presented by Dabdub and Seinfeld (1994) were based on a DTN approach to implementing the transport

in parallel. The advantages of DTN are that it is easy to code and is fully modular. DTN is easy to code since the programmer is not concerned about nodes with special cases. When implementing EA for a variable number of processors the programmer must be careful about corner nodes that have no neighbors. DTN is fully modular, like the sequential case, because the node performing the transport step has all the grid cell concentrations in case they are needed by a specific transport solver. On the other hand, DTN has two main disadvantages: it is slower than EA and it is not well balanced. To gain modularity DTN requires a greater number of internode communication than EA. The increase in communication traffic decreases performance results. In addition, if the number of rows or columns is smaller than the number of nodes available, DTN presents load imbalancing that affects performance further. Typical urban AQMs have less than 100 rows or columns. If there are 512 nodes available DTN will leave most of the nodes idle when performing the transport computations.

The Dynamical Balance, DYB, approach is an attempt to maintain the modularity of DTN while increasing its performance by decreasing the idle node time. Instead of having a predetermined node to perform the transport computations for all the species and layers of a particular row or column, DYB assigns the first N ground rows or columns to be performed in the first N nodes, where N is the number of available nodes. The next N rows or columns continue to be evenly distributed among the available N nodes. This distribution continues until all rows or columns have been assigned a transport node. Figure 2.10 illustrates the X-transport distribution of a computational domain with three layers containing five rows each among three nodes. The number assigned to each row corresponds to the node number in which the transport computations will be performed for the row. Note that, to maintain the load as balanced as possible, the distribution of rows or columns in layer m starts with the node number having fewer rows or columns assigned in layer $(m-1)$. As it can be seen, the DYB approach provides a well balanced load while maintaining the modularity of the code. Nevertheless, the programming required to implement DYB for the general case (any number of rows, columns, layers and nodes) requires more effort than EA or DTN. The main advantage of DYB is that it reduces the overall

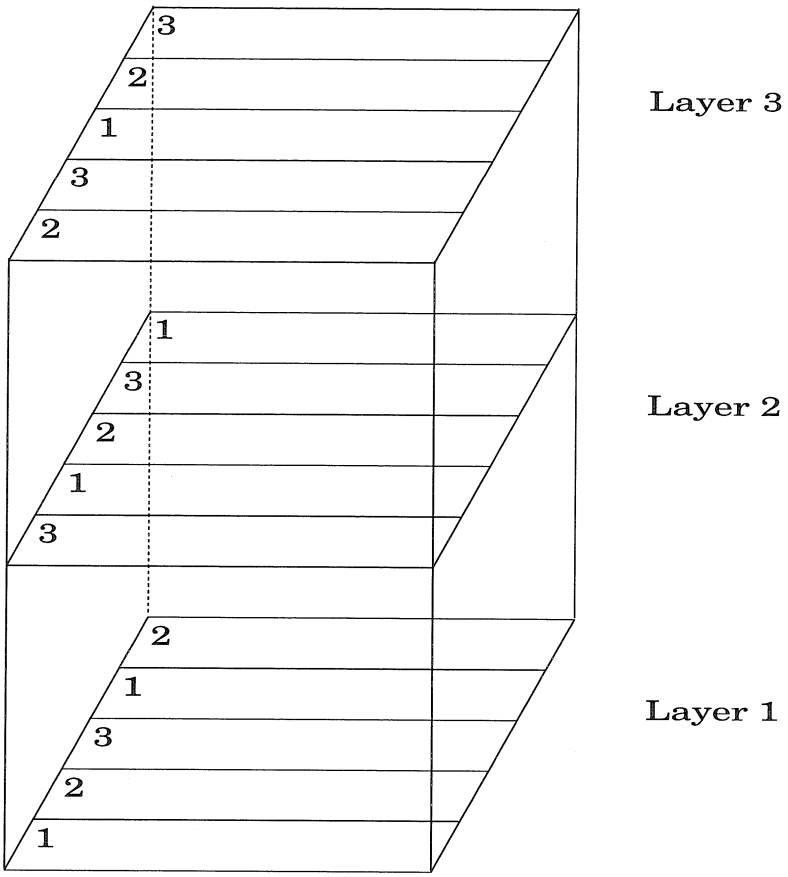


Figure 2.10: Example of row distribution to perform X-transport computations using a DYB approach to implement transport. The example is based on a three-layer domain containing five rows. The numbers shown in each row denote the processor number where the transport computations for such row are to be performed.

idle time of the nodes, especially in the massively parallel regime.

Figure 2.11 shows the CPU time versus the number of processors for a DTN and DYB approach to the parallel implementation of the CIT model using the Galerkin solver. The model was run on the Intel Touchstone Delta using NX as the message passing protocol. Figure 2.11 shows that similar times are required using the two different approaches to implement the algorithm in parallel. It is observed that the CPU time starts to flatten in the high number of processors regime. The reason for such behavior is that the workload required by the Galerkin solver is so light that it does not benefit from having extra nodes to perform it. Indeed, it is observed that the case for 256 nodes is slightly slower than the 128 nodes since at that point the nodes start to interfere with each other to efficiently carry out the computations.

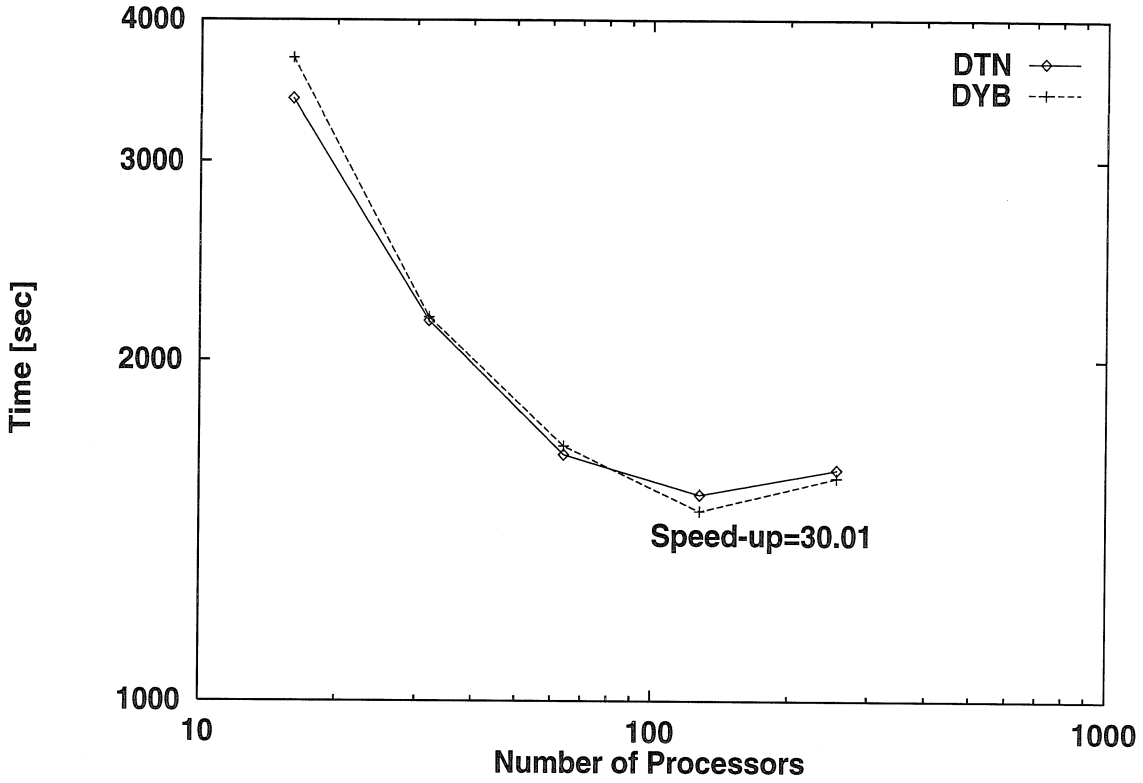


Figure 2.11: CPU time for a 24-hour simulation of the South Coast Air Basin versus number of nodes for the Galerkin transport implementation of the CIT model using DTN and DYB implementation strategies.

When using the Galerkin algorithm, the nodes spend a small fraction of their time performing the transport step. Therefore, the potential idle times induced by DTN are not significant. In general, when using any transport solver that it is not computationally intensive, the simple and easier to code DTN approach should be followed. The best speed-up observed for the Galerkin algorithm is 30.01 when using the DYB approach and 128 nodes.

Figure 2.12 shows the CPU time versus the number of processors for a DTN and DYB approach to the parallel implementation of the CIT model using the ASD solver. It is observed that the CPU also flattens in the high number of processors regime for the DTN approach. In this case, the flat region does not occur because the workload is light, but because the number of processors at that point exceeds the number of

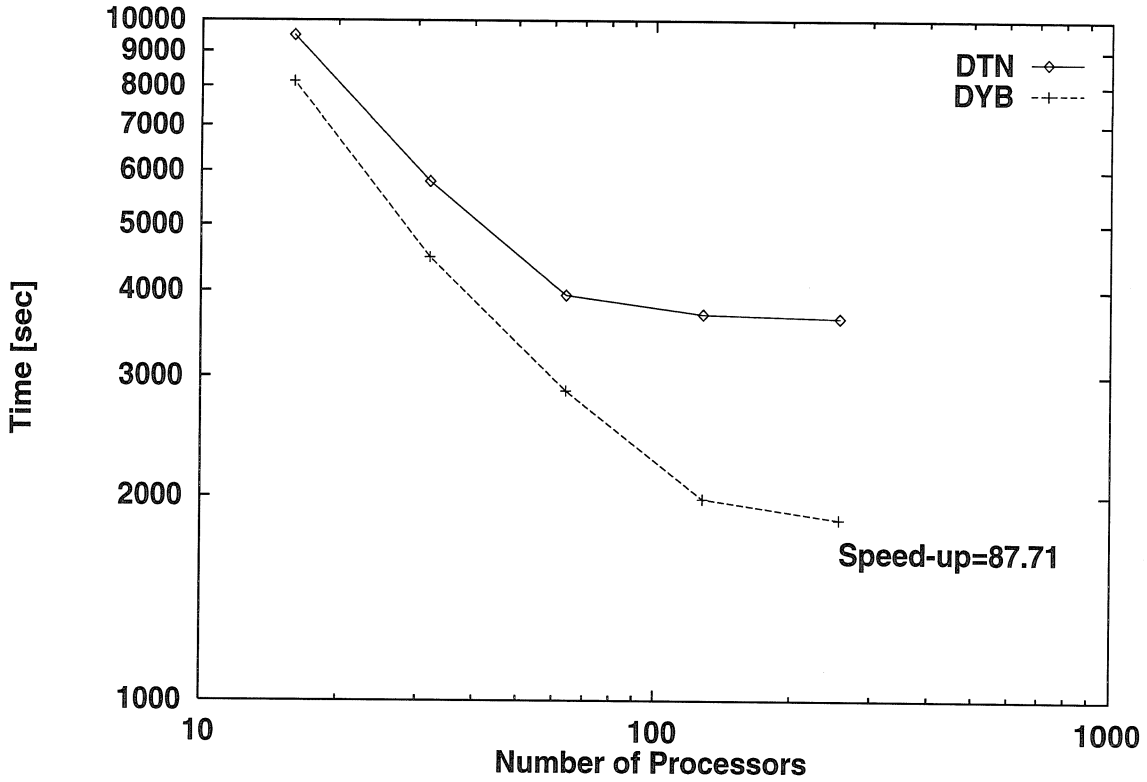


Figure 2.12: CPU time for a 24-hour simulation of the South Coast Air Basin versus number of nodes for the ASD transport implementation of the CIT model using DTN and DYN implementation strategies.

rows or columns in the computational grid. As a result, increasing the number of nodes for the DTN approach only increases their idle time. For the ASD method the DYB approach is clearly superior, especially in the massively parallel regime. The best speed-up factor observed for the ASD algorithm is 87.71 when using the DYB approach and 256 nodes. In general, when using a computationally intensive transport solver and performing the computations using a high number of nodes the DYB approach should be followed.

2.7 Conclusions

Our results confirm those of Chock that the ASD method stands as the best advection scheme tested based on its mass distribution ratio, mass conservation ratio, average absolute error, and peak preserving properties. Its high accuracy, however, is achieved at the price of a greater computation time that might be unacceptable on a sequential machine. However, with the use of parallel computers the computational time is substantially reduced. Indeed, it is well known that using a more CPU intensive algorithm for a given amount of interprocess communication results in greater speed-ups on a parallel environment. A typical 24-hour simulation using a Galerkin solver on the CIT model takes about 25 minutes using 256 nodes on the Touchstone Delta. The same run, using the more accurate ASD solver takes about 30 minutes using 256 nodes on the Touchstone Delta.

The second best algorithm tested was the finite element Galerkin scheme. If computation time is a constraint, as when running AQMs on sequential machines, the Galerkin algorithm is recommended. The Smolarkiewicz advection solver was found to produce poor results for the tests performed. Therefore, the use of this method in the UAM model should be reconsidered.

For Courant numbers relevant for air pollution modeling (greater than 0.1) the Forester filter performs the best for all the cases studied. The Forester filter parameters are found to have a significant impact on the results of the advection solver. In particular, we have been able to optimize the choice of the filter parameters used in the CIT model as: $K = 1$, $\mu = 0.1$, $m = 1$, and $n = 2$. Implementing these filter parameters into the CIT model leads to ozone peaks that are not present with the previous filter parameters. The ASD method confirms the validity of such peaks, and produces a greater concentration at the peaks as expected.

When implementing the advection solver in a parallel environment the EA approach should be used if speed is the greatest concern. DTN offers a fully modular approach that is easy to code, but it sacrifices some performance. DTN is recommended when the number of nodes available is smaller than the number of rows or

columns in the computational region of the AQM and a fast transport solver is being used such as Galerkin. Finally, the DYB approach is recommended when computing in the massively parallel regime while using intensive transport solvers such as ASD.

References

Bartnicki J. (1989) A simple filtering procedure for removing negative values from numerical solutions of the advection equation. *Environmental Software* **4,** No. 4 187-201.

Boris J. P. and Book D. L. (1973) Flux-corrected transport I. SHASTA, a fluid transport algorithm that works. *J. comp. Phys.* **11,** 38-69.

Bott A. (1989a) A positive definite advection scheme obtained by nonlinear renormalization of the advective fluxes. *Mon. Wea. Rev.* **117,** 1006-1015.

Bott A. (1989b) Reply *Mon. Wea. Rev.* **117,** 2633-2636.

Canosa J. and Gazdag J. (1976) The Korteweg-de Vries-Burger equation. *J. comp. Phys.* **23,** 393-403.

Carver M. B. and Hinds H. W. (1978) The method of lines and the advection equation. *Simulation* **31,** 59-69.

Chapman M. (1981) FRAM—Nonlinear damping algorithms for the continuity equation. *J. comp. Phys* **44,** 84-103.

Chock D. P. and Dunker A. M. (1983) A comparison of numerical methods for solving the advection equation. *Atmospheric Environment* **17,** 11-24.

Chock D. P. (1985) A comparison of numerical methods for solving the advection equation—II. *Atmospheric Environment* **19,** 571-586.

Chock D. P. (1991) A comparison of numerical methods for solving the advection equation—III. *Atmospheric Environment* **25A,** 853-871.

Dabdub D. and Seinfeld J. H. (1994) Air quality modeling on massively parallel computers *Atmospheric Environment,* in press.

Emde D. V. D. (1992) Solving conservation laws with parabolic and cubic splines *Mon. Wea. Rev.* **120,** 82-92

Engquist B., Lötstedt P., and Sjögreen B. (1989) Nonlinear filters for efficient shock computation. *Math. of Comp.* **52,** 509-537.

Forester C. K. (1977) Higher order monotonic convective difference schemes. *J. comp. Phys.* **23,** 1-22.

Fox G., Johnson M., Lyzenga G., Otto S., Salmon J. and Walker D. (1988) *Solving problems on concurrent processors Vol. I* Prentice-Hall, New Jersey.

Gazdag J. (1973) Numerical convective schemes based on accurate computation of space derivatives. *J. comp. Phys.* **13,** 100-113.

Gazdag J. and Canosa, J. (1974) Numerical solution of Fisher's equation. *J. Appl. Prob* **11,** 445-457.

Gazdag J. (1975) Numerical solution of the Vlasov equation with the Accurate Space Derivative Method. *J. comp. Phys.* **19,** 77-89.

Harley R. A., Russell A. G., McRae G. J., Cass G. R. and Seinfeld J. H. (1993) Photochemical modeling of the Southern California air quality study. *Environ. Sci. Technol.* **27,** 378-388.

Kahaner D., Moler C. and Nash S. (1989) *Numerical Methods and Software* Prentice-Hall Inc., Englewood Cliffs,New Jersey.

Hov Ø., Zlatev Z., Berkowicz R., Eliassen A. and Prahm L. P. (1989) Comparison of numerical techniques for use in air pollution models with nonlinear chemical reactions. *Atmospheric Environment* **23,** 967-983.

McRae G. J., Goodin W. R. and Seinfeld J. H. (1982) Numerical solution of the atmospheric diffusion equation for chemically reacting flows. *J. comp. Phys.* **45,** 1-42.

Morris R. E., Myers T. C. (1990) User's guide for the Urban Airshed Model—Volume I: User's manual for UAM (CB-IV). U.S. Environmental Protection Agency (EPA-450/4-90-007a).

Odman M. T and Russell A. G. (1993) A nonlinear filtering algorithm for multidimensional finite element pollutant advection schemes. *Atmospheric Environment* **27A,** 793-799.

Oran E. S. and Boris J. P. (1987) *Numerical Simulation of Reactive Flow.* Elsevier, New York.

Pepper D. W., Kern C. D. and Long P. E. (1979) Modeling the dispersion of atmospheric pollution using cubic splines and chapeau functions. *Atmospheric Environment* **13,** 223-237.

Pepper D. W. and Long P. E. (1978) A comparison of results using second-order moments with and without width correction to solve the advection equation. *J. appl. Met.* **17,** 228-233.

Raymond W. H. and Garder A. (1976) Selective damping in a Galerkin method for solving wave problems with variable grids. *Mon. Weath. Rev.* **104,** 1583-1590.

Roache P. J. (1978) A pseudo-spectral FFT technique for non-periodic problems. *J. comp. Phys.* **27,** 204-220.

Rood R. B. (1987) Numerical advection algorithms and their role in atmospheric transport and chemistry models. *Rev. Geophys.* **25,** 71-100.

Shapiro A. and Pinder G. F. (1981) Analysis of an upstream weighted collocation approximation to the transport equation. *J. comp. Phys.* **39,** 46-71.

Schere K. L. (1983) An evaluation of several numerical advection schemes *Atmospheric Environment* **17,** 1897-1907.

Schiesser, W. E. (1991) *The Numerical Method of Lines.* Academic Press, San Diego, California.

Sheih C. M. and Ludwig F. L. (1985) A comparison of numerical pseudodiffusion and atmospheric diffusion. *Atmospheric Environment* **19,** 1065-1068.

Smolarkiewicz P. K (1984) A simple positive definite advection scheme with small implicit diffusion. *Mon. Wea. Rev.* **111,** 479-486.

Tran K. T. and Mirabella V. A. (1991) A comparison of advection schemes in existing photochemical grid models. Air and Waste Management Assoc., No 91-66.2, 84th Annual meeting AWMA, Vancouver, BC, June 16-21.

Wengle H. and Seinfeld J. H. (1978) Pseudospectral solution of atmospheric diffusion problems. *J. comp. Phys.* **26,** 87-106.

Wengle H., Van den Bosch B. and Seinfeld J. H. (1978) Solution of atmospheric diffusion problems by pseudospectral and orthogonal collocation methods. *Atmospheric Environment* **12,** 1021-1032.

Yanenko N. N. (1971) *The Method of Fractional Steps.* Springer, New York.

Zalesak S. T. (1979) Fully multidimensional flux-corrected transport algorithms for fluids. *J. comp. Phys.* **31,** 335-362.

Appendix

Description of filters

Bartnicki Filter The Bartnicki filter (Bartnicki, 1989) uses the simplest redistribution strategy. It consists of adding all the negative mass and subtracting equal amounts from all positive values present in the distribution. The total amount of the mass subtracted from the positive values equals the total initial negative mass. The filter does not modify zero values of the distribution. Note that if all distribution values are non-negative the filter does not alter the distribution. It has been found that two iterations are sufficient to remove all the negative mass in the cases studied. While, this strategy guarantees the absence of negative values present after the filtering step, the filter does not sufficiently smooth the solution's positive oscillations.

Chapman Filter FRAM (Filtering Remedy and Methodology) was presented by Chapman (1981). The idea behind the filter is to introduce a strong nonlinear dissipation to the advection equation to dampen spurious oscillations. The FRAM algorithm can be outlined as follows:

(1) Calculate a provisional advanced time solution $\tilde{C}_i^{t+\Delta t}$ using a higher-order scheme.

(2) Calculate local bounds on the advanced time solution. In one dimension and constant wind velocity the bounds are:

$$C_{imin}^t = \min(C_{i-1}^t, C_i^t, C_{i+1}^t) \tag{2.21}$$

$$C_{imax}^t = \max(C_{i-1}^t, C_i^t, C_{i+1}^t). \tag{2.22}$$

(3) Introduce local diffusion where the provisional solution is not within the bounds. Conserving C, the local diffusion is introduced as:

$$C_i^{t+\Delta t} = \tilde{C}_i^{t+\Delta t} + (\alpha_i^t + \alpha_{i+1}^t)(C_{i+1}^t - C_i^t) + (\alpha_i^t + \alpha_{i-1}^t)(C_{i-1}^t - C_i^t), \tag{2.23}$$

where α_i^t can be thought of as a diffusion coefficient that is computed using the provisional concentrations bounds computed in (22) and (23).

Engquist Filter Engquist *et al.* (1988) have proposed a series of nonlinear filters of differing complexity. The filters do not introduce any diffusion to the advection equation. The basic algorithm behind all the filters proposed is:

(1) Scan $C_i^{t+\Delta t}$ and correct the i-values that are local maxima or local minima. Corrections are made by decreasing the maxima and by increasing the minima.

(2) If point i is to be corrected, the same correction must be subtracted from point $i-1$ or $i+1$, whichever has the greatest distance to $C_i^{t+\Delta t}$.

(3) No value may be corrected so that it is a new extremum.

The filter implemented in this study is algorithm 2.4 described by Engquist *et al.* (1988). It makes the smallest correction to the distribution while still being TVD.

Forester Filter To dampen the spurious oscillations the Forester filter (Forester, 1977) introduces a local diffusion to ensure that local extrema are separated by $2n$ mesh intervals. The filter uses n as a parameter, as well as m, K, and μ. K is the number of filtering iterations and μ is a dimensionless diffusion coefficient. The parameters m and n determine the noise wavelength to be filtered. These free parameters are problem dependent and they might significantly affect the performance of the filter. The Forester filter is described by the following iterative scheme:

$$C_i^{k+1} = C_i^k + \frac{\mu}{2}[(C_{i+1} - C_i)(\psi_i + \psi_{i+1}) - (C_i - C_{i-1})(\psi_i + \psi_{i-1})]^k, \qquad (2.24)$$

where C_i^{k+1} is the value of C_i after k iterations of the filter. The values of ψ_i are either 0 or 1 and determine the points at which smoothing occurs. They are computed from the filter parameters m and n.

Chapter 3

Extrapolation Techniques Used in the Solution of Stiff ODEs Associated with Chemical Kinetics of Air Quality Models

Donald Dabdub and John H. Seinfeld.

Atmospheric Environment, **29**, (1995) 403–410.

The solution of chemical kinetics is generally the most computationally intensive step in atmospheric air quality models. The incorporation of ever more complex chemical mechanisms and physico-chemical phenomena into these models stimulates the search for more accurate and efficient numerical ODE integration methods. We report here on a new method based on Richardson extrapolation to solve the chemical kinetics in air quality models. The Extrapolation method presents high accuracy consistently for wide ranges of ROG/NO_x ratios. The method is robust during sunrise and sunset transitions, when the rate of change of concentrations of a number of photochemically driven species is the greatest. In addition, the Extrapolation algorithm is one of the most efficient computationally tested.

3.1 Introduction

In three-dimensional Eulerian Air Quality Models (AQMs) the computation of the rate of chemical reaction is the most intensive calculation component, requiring 75-90% of the total CPU time (Shin and Carmichael, 1992; Saylor and Fernandes, 1993; Dabdub and Seinfeld, 1994). To compute the rate of chemical reaction one must essentially solve a system of stiff nonlinear ordinary differential equations (ODEs) of the form

$$\frac{dc_i}{dt} = P_i(\mathbf{c}, t) - L_i(\mathbf{c}, t)c_i, \tag{3.1}$$

where P_i and $L_i c_i$ are the production and loss rate of species i, respectively, and $\mathbf{c}$ is the vector of concentrations. The chemistry in urban and regional ozone and acid deposition AQMs is becoming more comprehensive and complex as these models continue to be refined and developed. The CIT urban photochemical model, for example, incorporates a modified version of the LCC chemical mechanism (Lurmann *et al.,* 1987), consisting of 106 reactions involving 36 chemical species (Harley *et al.,* 1993). The Urban Airshed Model (UAM) employs the Carbon Bond Mechanism version IV (CB-IV) (Morris and Myers, 1990) containing 87 reactions and 38 species. The Regional Acid Deposition Model (RADM) chemical mechanism consists of 157 reactions involving 59 chemical species (Stockwell *et al.,* 1990).

The numerical solution of stiff ODEs has been the subject of considerable attention in the numerical analysis literature (Gear, 1971; Lapidus and Seinfeld, 1971; Enright *et al.,*, 1975; Hairer, 1987). There exist a number of general purpose software packages to obtain accurate solutions to stiff ODE systems. Byrne and Hindmarsh (1987) compare some of the most popular packages. This study focuses on the Livermore Solver for Ordinary Differential Equations (LSODE), The Variable-Coefficient Ordinary Differential Equation Package (VODPK), the Quasi-Steady State (QSSA) method, the Hybrid method, and the Extrapolation method which is developed here. LSODE (Hindmarsh, 1980) is generally regarded as one of the most accurate routines available and is frequently used as a benchmark to evaluate other methods. In LSODE the Jacobian matrix of the system must be computed and a set of algebraic

equations must be solved at each time step, relatively time-consuming operations. To overcome the time requirements of LSODE other methods have been introduced that are specifically designed to solve the ODEs resulting from chemical kinetics in a more rapid manner. VODPK is an alternative that might offer dramatic storage and CPU time improvements in comparison with elimination solvers. It uses Krylov subspace projection methods and the Nordisieck formulation of backward differentiation formulas (Brown and Hindmarsh, 1989). The implementation of VODPK studied here uses Krylov projections with unpreconditioned iterations. Two of the most widely used integrators, as applied to solving chemical kinetics associated with atmospheric chemistry, are the the QSSA method (Hesstvedt *et al.,* 1978) and the Hybrid method (Young and Boris, 1977). Table 3.1 outlines the characteristics of the methods compared in this study.

The QSSA method as proposed by Hesstvedt *et al.* (1978) uses the following formulas for advancing the solution over a time step Δt:

(a) If $L_i(\mathbf{c})\Delta t < 0.01$, where $L_i(\mathbf{c})c_i$ is the species loss rate as defined in Equation (3.1), the equations are non-stiff and are advanced in time by a simple Euler step

$$c_i(t+\Delta t) = c_i(t) + \frac{dc_i}{dt}\Delta t. \tag{3.2}$$

(b) If $10 \geq L_i(\mathbf{c})\Delta t \geq 0.01$, the equations are deemed stiff and are advanced in time using

$$c_i(t+\Delta t) = \frac{P_i(\mathbf{c})}{L_i(\mathbf{c})} + \left(c_i(t) - \frac{P_i(\mathbf{c})}{L_i(\mathbf{c})}\right)\exp(-L_i(\mathbf{c})\Delta t), \tag{3.3}$$

which comes from the analytical solution of Equation (3.1) assuming that P_i and L_i are constants.

(c) If $L_i(\mathbf{c})\Delta t > 10$, the species i is considered to be in a quasi-steady state and

$$c_i(t+\Delta t) = \frac{P_i(\mathbf{c})}{L_i(\mathbf{c})}. \tag{3.4}$$

The Hybrid method (Young and Boris, 1977) uses the following predictor-multicorrector algorithm. *(a)* If $L_i(\mathbf{c})\Delta t < 1$, the equations are considered non-stiff and

Table 3.1: ODE integration methods for chemical kinetics of Air Quality Models.

Method	Reference	Characteristics
LSODE	Hindmarsh (1980)	General stiff and nonstiff solver. Uses Adams methods (predictor-corrector) in the nonstiff case, and backward differentiation methods for the stiff case. Treats Jacobian matrix as a full or banded matrix. The linear systems that arise are solved by direct methods (LU factor/solve).
VODPK	Brown and Hindmarsh (1989)	General stiff and nonstiff solver. Uses Adams methods in the nonstiff case, and backward differentiation methods for the stiff case. Uses Krylov subspace iterative methods with right, left or no preconditioners and scaling.
QSSA	Hesstvedt, *et al.* (1978)	Stiff solver specifically designed for chemical kinetics problems. Uses analytical characteristics of rate equations for integration. Fixed time step. Some stability problems in long time simulations. Equations (3.2)-(3.4) in text.
Hybrid	Young and Boris (1977)	Stiff solver specifically designed for chemical kinetics problems. Uses asymptotic approximations to rate equations for integration. Variable time step. Predictor-multi-corrector algorithm. Equations (3.5)-(3.8) in text
Extrapolation	This work	A general solver that uses Richardson extrapolation to improve accuracy over single time step solution. To solve chemical kinetics problems the algorithm proposed in this work uses a modified QSSA scheme. Equations (3.2)-(3.4) and (3.12)-(3.15) in text.

are integrated with the following predictor (p) - corrector (c) method:

$$c_i^p(t+\Delta t) = c_i(t) + \frac{dc_i}{dt}\Delta t \tag{3.5}$$

$$c_i^c(t+\Delta t) = c_i(t) + (\frac{dc_i}{dt} + \frac{dc_i^p}{dt})\frac{\Delta t}{2}. \tag{3.6}$$

(b) If $L_i(\mathbf{c})\Delta t \geq 1$, the equations are considered stiff and are integrated with the following asymptotic predictor and corrector formulas:

$$c_i^p(t+\Delta t) = \frac{c_i(t)(2/L_i(\mathbf{c}) - \Delta t) + 2\Delta t P_i(\mathbf{c})/L_i(\mathbf{c})}{2/L_i(\mathbf{c}) + \Delta t} \tag{3.7}$$

$$c_i^c(t+\Delta t) = \frac{c_i(t)(\psi_i - \Delta t) + \Delta t(P_i(\mathbf{c}) + P_i(\mathbf{c}^p))\psi_i/2}{\psi_i + \Delta t}, \tag{3.8}$$

where $\psi_i = 1/L_i(\mathbf{c}) + 1/L_i(\mathbf{c}^p)$, and c_i^p and c_i^c are the predicted and corrected concentrations, respectively.

The QSSA and Hybrid methods applied to atmospheric chemistry have been compared to LSODE by Odman *et al.* (1992), who determined that the Hybrid method is more robust and accurate than the QSSA scheme, even though QSSA is a faster algorithm. Because of stability problems associated with simulating day-to-night transitions (Odman, 1992), the QSSA is not a viable method to be used as the integrator in photochemical AQMs.

The critical need for ODE integration algorithms for AQMs is high efficiency while maintaining accuracy and robustness. Although the Hybrid method has been proved to be a robust integration method for atmospheric chemistry, its accuracy is not as good as one might seek. In addition, the incorporation of ever more complex chemical mechanisms and physical phenomena, such as aerosol processes, into AQMs stimulates the search for even more accurate and efficient integration methods. This paper presents a new chemical integrator for AQMs based on extrapolation techniques. The performance of the Extrapolation method is compared to that of the LSODE, QSSA, and Hybrid method using a test case of organic/NO_x chemistry as well as a

full implementation of the method in the CIT model.

3.2 Description of the Method

Extrapolation methods, or so called Richardson extrapolation, consist of solving a system of ODEs repeatedly using the same scheme, but with ever decreasing time steps, and then combining the results of the solutions to obtain a result that is more accurate than any of the individual solutions (Lapidus and Seinfeld, 1971; Stoer and Burlisch, 1980). For example, in a two time step implementation, the ODEs are first solved with scheme $\mathcal{L}_h$, with time step h. This produces

$$\mathcal{L}_h\{\mathbf{c}(t)\} = \mathbf{c}(t+\Delta t; h) = \mathbf{c}(t+\Delta t) + \mathbf{R}_m(\mathbf{c})h^m + O(h^{m+1}), \tag{3.9}$$

where $\mathbf{c}(t+\Delta t; h)$ is the approximation of the species concentrations, $\mathbf{c}(t+\Delta t)$ is the true solution to the differential equations, m is the order of the scheme, and $\mathbf{R}_m$ and O represent the error inherent in $\mathcal{L}_h$ of order h^m and h^{m+1}, respectively . The equations are solved again with the same scheme but with a different time step k. This produces

$$\mathcal{L}_k\{\mathbf{c}(t)\} = \mathbf{c}(t+\Delta t; k) = \mathbf{c}(t+\Delta t) + \mathbf{R}_m(\mathbf{c})k^m + O(k^{m+1}). \tag{3.10}$$

Then, Equations (3.9) and (3.10) are combined using Richardson extrapolation to yield the approximation,

$$\mathbf{c}(t+\Delta t; h, k) = \frac{k^m\mathbf{c}(t+\Delta t; h) - h^m\mathbf{c}(t+\Delta t; k)}{k^m - h^m} \tag{3.11}$$

which is accurate to order $h^{m+2} + k^{m+2}$

While any scheme can be selected as the basis for the Extrapolation method, since $\mathcal{L}$ must be called at least twice, its speed is central to the efficiency of the method. We introduce here a version of the QSSA method that has been modified by including a corrector step as well as a convergence test as the basic scheme for the Extrapolation

method described by Equations (3.9)-(3.11).

The modified QSSA first predicts the concentration of species i, $c_i^p(t+\Delta t;\Delta t)$ using the QSSA method, Equations (3.2)-(3.4). After the prediction step is completed P, L, and dc_i/dt are re-evaluated using the predicted concentration $c_i^p(t+\Delta t;\Delta t)$. Then, the predicted concentrations are corrected. The correction step is introduced in QSSA to increase the robustness of the method since it has been demonstrated that in long time simulations QSSA presents instability problems (Odman *et al.,* 1992).

The corrector step of the modified QSSA is performed in the following manner:

(a) If $L_i(\mathbf{c}^p)\Delta t < 0.01$, the equations are non-stiff and are corrected using the trapezoidal rule

$$c_i^c(t+\Delta t;\Delta t) = c_i(t) + (\frac{dc_i}{dt} + \frac{dc_i^p}{dt})\frac{\Delta t}{2}. \tag{3.12}$$

(b) If $10 \geq L_i(\mathbf{c}^p)\Delta t \geq 0.01$, the equations are corrected using

$$c_i^c(t+\Delta t;\Delta t) = \psi_i + (c_i(t) - \psi_i)\exp\left[-\left(\frac{1}{L_i(\mathbf{c})} + \frac{1}{L_i(\mathbf{c}^p)}\right)\frac{\Delta t}{2}\right]. \tag{3.13}$$

(c) If $10 < L_i(\mathbf{c}^p)\Delta t$, the following numerical approximation is used

$$c_i^c(t+\Delta t;\Delta t) = \psi_i, \tag{3.14}$$

where

$$\psi_i = \frac{1}{4}\left(P_i(\mathbf{c}) + P_i(\mathbf{c}^p)\right)\left(\frac{1}{L_i(\mathbf{c})} + \frac{1}{L_i(\mathbf{c}^p)}\right). \tag{3.15}$$

Note that: (1) If the modified QSSA is used by itself (not as part of Extrapolation) then the convergence of the corrector must be checked by assuring that $|c_i^p - c_i^c|/c_i^c < \epsilon$ for all i, where ϵ is a small number provided by the user. If convergence is not achieved the time step is decreased. On the other hand, if the modified QSSA is used as part of Extrapolation then h and k are kept constant. (2) All the species must be determined to be stiff or non-stiff for every time step. (3) There is no steady-state assumption that requires equating first-order time derivatives of steady-state species to zero and solving the resulting nonlinear algebraic equations.

Table 3.2: Initial mixing ratios in parts-per-billion (ppb) for the three single cell cases.

Species		Case 1 $\alpha = 1$	Case 2 $\alpha = 10$	Case 3 $\alpha = 100$
NO	$\alpha \times 7.5; \alpha = 1, 2, \ldots 100$	7.5	75	750
NO_2	$\alpha \times 2.5; \alpha = 1, 2, \ldots 100$	2.5	25	250
O_3	0.01	0.01	0.01	0.01
CO	1000	1000	1000	1000
HCHO	30	30	30	30
ALD2	10	10	10	10
ALKA	82.7	82.7	82.7	82.7
ETHE	15	15	15	15
ALKE	29.4	29.4	29.4	29.4
TOLU	22.9	22.9	22.9	22.9
AROM	11.9	11.9	11.9	11.9
H_2O	2×10^7	2×10^7	2×10^7	2×10^7

3.3 Numerical Experiments

In this section we compare the speed and accuracy of the Extrapolation method against the LSODE, VODPK, QSSA, and Hybrid algorithms. The comparison is based on two test cases: (1) a single cell simulation used by Odman *et al.* (1992) based on that of Lurmann *et al.,* (1987); (2) a full implementation in the CIT model.

3.3.1 Single Cell Test

The test case consists of simulating the 8-hr photooxidation of a complex organic/NO_x mixture inside a single cell at a constant temperature, 298 K. Table 3.2 lists the initial mixing ratios for the cases studied. The cases differ only in the NO and NO_2 initial concentration by a factor of α so as to encompass (ROG)/NO_x ratios between 0.2 and 20.2. The photolytic reaction rates are calculated at a constant zenith angle of 0°. The modified LCC mechanism was used with the initial conditions described by Lurmann *et al.* (1987). Table 3.3 lists all the species in the modified LCC mechanism used. There are no emissions or deposition.

The integrators used are LSODE, VODPK, QSSA, Hybrid, and Extrapolation

Table 3.3: Species in chemical kinetic mechanism used in this study.

Species Number	Species Name	Species Number	Species Name	Species Number	Species Name
1	NO	13	MEK	25	CRES
2	NO2	14	MGLY	26	NPHE
3	O3	15	PAN	27	H2O2
4	HONO	16	RO2	28	MEOH
5	HNO3	17	MCO3	29	NH3
6	HNO4	18	ALKN	30	NIT
7	N2O5	19	ALKA	31	ISOP
8	NO3	20	ETHE	32	EOTH
9	HO2	21	ALKE	33	MTBE
10	CO	22	TOLU	34	SO2
11	HCHO	23	AROM	35	SO3
12	ALD2	24	DIAL		

method. LSODE is called with the following error control parameters: `RTOL=1.0d-6` and `ATOL=1.0d-9`. QSSA is implemented with a fixed time step of 30 s, the value suggested by Hesstvedt *et al.*, (1977) as appropriate for most simulations of photochemical air pollution. The Extrapolation method is implemented using the modified QSSA as the basic scheme with constant time steps of 60 s and 30 s for the extrapolation. All the integrators were called at five minute intervals.

When implementing the Extrapolation method, the following two questions must be addressed: (1) How to compute the order of accuracy m? (2) How often should extrapolation be performed? We have used the following approach to address these questions. First, m is computed numerically. This is done by calling the modified QSSA scheme with a constant step of 3 s from $t = 0$ to $t = 5$ min to obtain an accurate value for c_i. The accurate value of c_i is used as the left-hand-side of Equation (3.11) to solve for m. The value of m associated with each species is then used through the simulation for the first 100 min. Since m changes slightly with time due to the fact that the degree of stiffness of the equations is also changing as the simulation progresses, at each 100 minutes it is recomputed. The Richardson extrapolation is performed every five min. This time corresponds to the time interval between integrator calls. The extrapolated values are then used as the initial conditions for

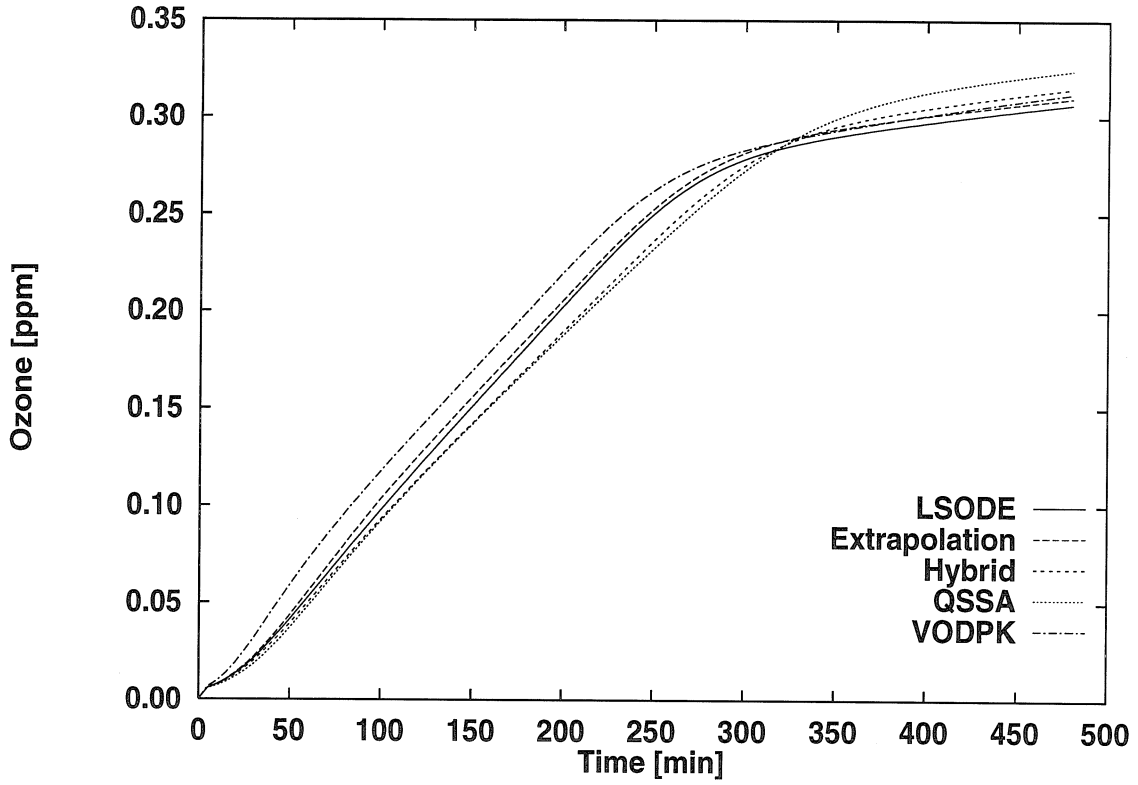


Figure 3.1: Ozone mixing ratios from single cell simulation of case 1 using LSODE, QSSA, VODPK, Hybrid, and Extrapolation algorithms.

the next integrator calls.

Figure 3.1 shows the ozone (O_3) mixing ratio in parts-per-billion (ppb) for case 1 using the four integrators mentioned. Odman *et al.* (1992) presented similar results using the LSODE, QSSA, and Hybrid methods. While the QSSA results deviate most from those obtained by LSODE, and the Extrapolation results agree most closely with those of LSODE, none of the four integrators yields major differences for ozone in this particular test.

Ozone, however, is not the most sensitive indicator of the performance of an integration method for atmospheric chemistry. Rather, free radical species such as the hydroperoxyl (HO_2) and nitrate (NO_3) radicals should reflect more strongly differences in the performance of numerical integration routines. Tables 3.4-3.6 show the maximum relative error for VODPK, QSSA, Hybrid, and Extrapolation methods

for each of the three simulations. Tables 3.4-3.6 also show the average normalized relative error, which is the computed using the maximum relative errors for all the species during the entire time interval. It can also be noted that the performance of QSSA varies among all the cases. The accuracy of QSSA could be improved by taking smaller time steps, which would decrease the efficiency of the method. The average error of VODPK is the greatest of all the methods compared for all the cases studied. The reason for this behavior is that to obtain the best results with VODPK, one would need to implement right and/or left preconditioners. The selection of an effective preconditioner is problem dependent and there is no method that can be followed to obtain the optimum preconditioner for a given problem. The average errors presented in Tables 3.4-3.6 indicate that Hybrid and Extrapolation are the most accurate methods. Figure 3.2 shows the average error of the Hybrid and Extrapolation methods for 100 different mixing ratios. Except for the very first point (Case 1) the average error presented by Extrapolation is smaller than Hybrid algorithm.

As expected, the QSSA algorithm is the fastest of all those tested but with low accuracy. VODPK is the least accurate integrator tested. LSODE, VODPK, Extrapolation, and Hybrid integrators are 8.09, 5.90, 2.27, and 2.09 times slower than QSSA, respectively. These timing results for LSODE, Hybrid, and QSSA agree with those reported by Odman *et al.,* (1992). The Hybrid and Extrapolation integrators are similar in speed, however the Extrapolation method is superior in accuracy for the test case studied. If a higher initial mixing ration for ozone is used, the relative performance of the integrators is not affected.

3.3.2 Three-Dimensional Model Test

This section describes a comparison of the Extrapolation, LSODE, and Hybrid methods when implemented in the three-dimensional CIT model. The QSSA integrator is not discussed further since it is unstable under the conditions tested. VODPK is not discussed further since it presented the lowest accuracy. The integrators were used in the CIT model to simulate photochemical smog in the South Coast Air Basin during

Table 3.4: Normalized maximum error relative to LSODE for VODPK, QSSA, Hybrid, and Extrapolation methods for Case 1 simulation of 480 minutes duration.

	VODPK	**QSSA**	**HYBRID**	**EXTRAPL**
Species Number	max Error	max Error	max Error	max Error
1	0.2326	0.3817	0.0498	0.1123
2	0.4783	0.3244	0.0256	0.0430
3	0.1282	0.1737	0.0319	0.0663
4	0.4282	0.2069	0.0361	0.0310
5	0.2748	0.2801	0.0449	0.1003
6	0.4827	0.4575	0.0735	0.1588
7	0.7790	0.5999	0.1312	0.2812
8	0.5754	0.5357	0.1056	0.2181
9	0.2740	0.3433	0.0525	0.1121
10	0.0068	0.0026	0.0001	0.0003
11	0.2558	0.0420	0.0031	0.0072
12	0.5493	0.0854	0.0127	0.0289
13	0.4278	0.2036	0.0296	0.0683
14	1.3955	0.3255	0.0478	0.1091
15	0.2064	0.3422	0.0633	0.1359
16	0.4096	0.3485	0.0526	0.1136
17	0.8642	0.1391	0.0273	0.0573
18	0.5583	0.2018	0.0291	0.0673
19	0.0035	0.0330	0.0007	0.0025
20	0.1189	0.0766	0.0014	0.0057
21	0.8718	0.3465	0.0069	0.0296
22	0.0774	0.0451	0.0009	0.0034
23	0.4622	0.2770	0.0051	0.0199
24	0.2407	0.3134	0.0450	0.1044
25	0.2760	0.3210	0.0470	0.1074
26	0.1986	0.2634	0.0263	0.0643
27	0.2993	0.0492	0.0256	0.0437
28	0.0000	0.0000	0.0000	0.0000
29	0.0000	0.0000	0.0000	0.0000
30	0.0000	0.0000	0.0000	0.0000
31	0.0000	0.0000	0.0000	0.0000
32	0.0000	0.0000	0.0000	0.0000
33	0.0000	0.0000	0.0000	0.0000
34	0.0000	0.0000	0.0000	0.0000
35	0.0000	0.0000	0.0000	0.0000
Average Error	0.3107	0.1920	0.0279	0.0598

Table 3.5: Normalized maximum error relative to LSODE for VODPK, QSSA, Hybrid, and Extrapolation methods for Case 2 simulation of 480 minutes duration.

	VODPK	**QSSA**	**HYBRID**	**EXTRAPL**
Species Number	max Error	max Error	max Error	max Error
1	0.2813	0.2879	0.0984	0.0094
2	0.3443	0.7481	0.7997	0.0798
3	0.4551	0.1849	0.0639	0.0103
4	0.3041	0.4902	0.1737	0.0512
5	0.1980	0.2122	0.0807	0.0316
6	0.8821	0.6870	0.6699	0.1019
7	1.0593	0.9037	1.6993	0.1916
8	0.6198	0.6340	0.5066	0.0819
9	0.2674	1.2471	0.3563	0.0432
10	0.0090	0.0021	0.0008	0.0007
11	0.2178	0.0765	0.0407	0.0123
12	0.7106	0.1333	0.0518	0.0140
13	0.9215	0.0803	0.0485	0.0330
14	2.3878	0.1240	0.0822	0.0409
15	0.9266	0.1701	0.0822	0.0363
16	0.5682	1.7878	0.4093	0.0577
17	0.7615	2.5013	0.4501	0.0998
18	1.1069	0.1187	0.0473	0.0340
19	0.1519	0.1159	0.0264	0.0088
20	0.1796	0.2320	0.0548	0.0166
21	0.9292	0.0945	0.0908	0.0313
22	0.1380	0.1607	0.0360	0.0120
23	0.7891	0.0774	0.1196	0.0494
24	0.4758	0.2323	0.1650	0.0538
25	0.7206	0.2159	0.1214	0.0403
26	1.1284	0.5599	0.1469	0.0358
27	0.4201	1.9260	0.5426	0.0729
28	0.0000	0.0000	0.0000	0.0000
29	0.0000	0.0000	0.0000	0.0000
30	0.0000	0.0000	0.0000	0.0000
31	0.0000	0.0000	0.0000	0.0000
32	0.0000	0.0000	0.0000	0.0000
33	0.0000	0.0000	0.0000	0.0000
34	0.0000	0.0000	0.0000	0.0000
35	0.0000	0.0000	0.0000	0.0000
Average Error	0.4844	0.4001	0.1990	0.0357

Table 3.6: Normalized maximum error relative to LSODE for VODPK, QSSA, Hybrid, and Extrapolation methods for Case 3 simulation of 480 minutes duration.

	VODPK	**QSSA**	**HYBRID**	**EXTRAPL**
Species Number	max Error	max Error	max Error	max Error
1	0.0707	0.0162	0.0109	0.0074
2	0.0952	0.0324	0.0230	0.0220
3	0.1733	0.0494	0.0334	0.0290
4	0.1690	0.0288	0.0270	0.0219
5	0.2173	0.0291	0.0083	0.0045
6	0.4996	0.0564	0.0202	0.0175
7	0.4844	0.1331	0.0855	0.0767
8	0.4017	0.0971	0.0982	0.0557
9	0.4360	0.0212	0.3161	0.0065
10	0.0044	0.0002	0.0002	0.0001
11	0.0735	0.0070	0.0087	0.0075
12	0.6996	0.0041	0.0019	0.0009
13	2.2658	0.0088	0.0097	0.0081
14	2.6767	0.0142	0.0130	0.0118
15	1.1724	0.0560	0.0229	0.0121
16	0.6988	0.0245	0.1725	0.0099
17	1.0652	0.0179	0.0533	0.0120
18	1.3820	0.0144	0.0135	0.0111
19	0.0596	0.0017	0.0018	0.0015
20	0.0291	0.0026	0.0028	0.0023
21	0.8641	0.0028	0.0014	0.0007
22	0.0195	0.0023	0.0024	0.0020
23	0.6086	0.0128	0.0134	0.0107
24	0.5991	0.0159	0.0163	0.0146
25	0.8453	0.0108	0.0067	0.0049
26	1.0411	0.0134	0.0099	0.0073
27	1.0115	0.0352	0.0090	0.0057
28	0.0000	0.0000	0.0000	0.0000
29	0.0000	0.0000	0.0000	0.0000
30	0.0000	0.0000	0.0000	0.0000
31	0.0000	0.0000	0.0000	0.0000
32	0.0000	0.0000	0.0000	0.0000
33	0.0000	0.0000	0.0000	0.0000
34	0.0000	0.0000	0.0000	0.0000
35	0.0000	0.0000	0.0000	0.0000
Average Error	0.5047	0.0202	0.0281	0.0104

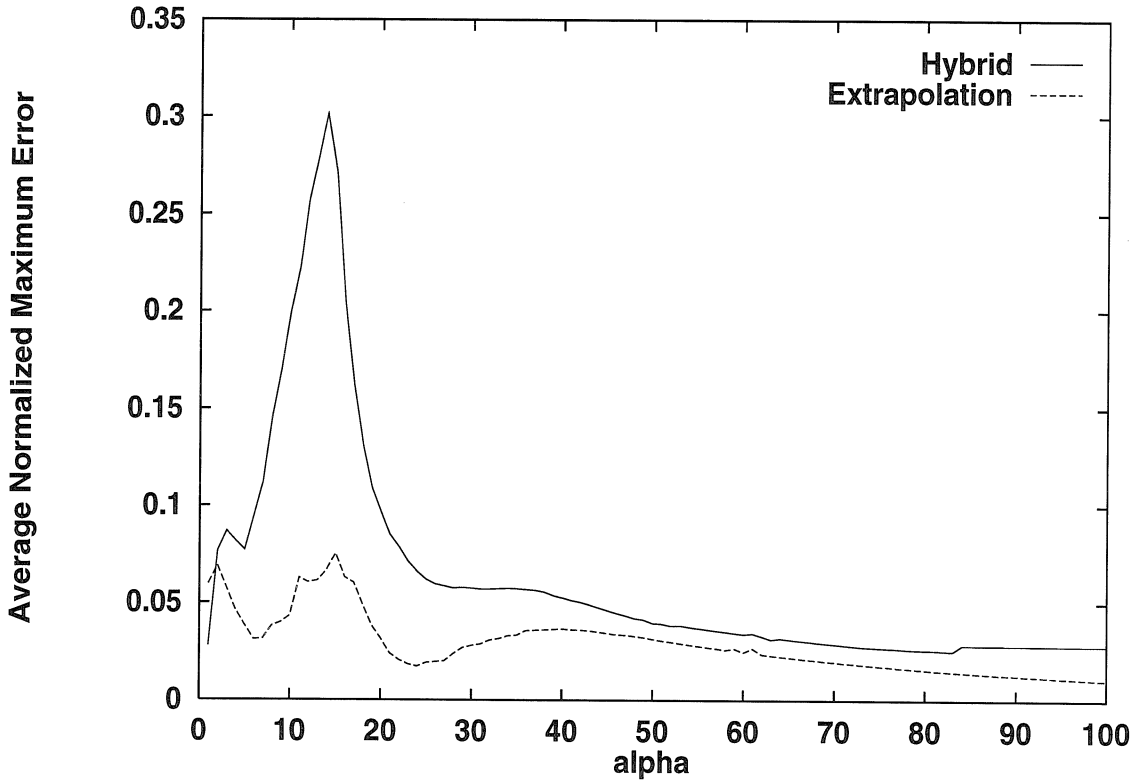


Figure 3.2: Average maximum relative errors computed over all species using Hybrid and Extrapolation algorithms for the 100 different ROG/NO_x ratios as described in Table 3.2.

the August 27-28, 1987 episode. A simulation that includes daytime and nighttime conditions tests the robustness of an integration method, since the night-to-day and day-to-night transitions lead to rapidly varying concentrations of a number of photochemically driven species. This test comparess the performance of the integrators interacting with other physical phenomena present in typical air quality models.

The LSODE and Hybrid integrators were implemented as described in the single cell test. The calculation of m in the Extrapolation method was carried out as described previously but only for a single vertical column of the model grid. The computed value of m is then used for all columns during the length of the chemistry step specified by the model. The Richardson extrapolation is performed every five minutes as in the single cell test case described previously.

To test the accuracy of the integrators the averaged normalized absolute errors were computed:

$$\text{error}_{i,x,y}(t) = \frac{|c^*_{i,x,y}(t) - c_{i,x,y}(t)|}{c^*_{i,x,y}(t)} \times 100 \tag{3.16}$$

where $c^*_{i,x,y}(t)$ and $c_{i,x,y}(t)$ are the predicted concentrations of species i at vertical column located at ground-level location x, y at time t by LSODE and the integrator being tested, respectively. The averaged normalized error is reported, instead of the average maximum error, since some of the mixing ratios predicted are below machine accuracy.

The error was determined for the Hybrid and Extrapolation integrators for each hour of a 24-hr simulation averaged over all the species and columns. Both integrators produce average errors on the order of 0.005% with no noticeably greater accuracy of one algorithm over the other. Thus a typical South Coast Air Basin episode presents ROG/NO_x ratios that do not affect the performance of the integrator. Extrapolation and LSODE were 1.13 and 14.33 times slower, respectively, than the Hybrid method.

3.4 Conclusion

This paper presents a new method based on Richardson extrapolation to solve the ODEs associated with chemical kinetics of reactive-flow problems, such as the atmospheric chemical mechanism of air quality models. The Extrapolation method is a general technique that can be used to solve any system of ODEs using different schemes as the basis for the extrapolation procedure. The basis method selected here is a version of the QSSA method (Hesstvedt *et al.,* 1978) modified by adding a corrector step and by checking the convergence of the corrector.

Numerical experiments show that the Extrapolation method consistently achieves high accuracy for wide ranges of ROG/NO_x ratios. The speed of the Extrapolation methods compares to that of the Hybrid method. In addition, there were no stability problems observed for the different single cell tests as well as the three-dimensional model tests. Overall the use of Extrapolation methods in air pollution appears to be a promising alternative to available methods.

References

Brown P. N. and Hindmarsh A. C. (1989) Reduced storage matrix methods in stiff ODE systems. *J. Appl. Math. Comput.* **31,** 40-91.

Byrne G. D. and Hindmarsh A. C. (1987) Stiff ODE solvers: a review of current and coming attractions. *J. Comp. Phys.* **70,** 1-62.

Byrne G. D. (1992) Pragmatic experiments with Krylov methods in the stiff ODE Setting. Numerical ODEs, Cash J. and Duff I., eds., Oxford University Press, London.

Dabdub D. and Seinfeld J. H. (1994) Air quality modeling on Massively Parallel Computers *Atmospheric Environment.* **28,** 1679-1687.

Enright W. H., Hull T. E. and Lindberg B. (1975) Comparing numerical methods for stiff systems of ODEs. *BIT* **15,** 10-48.

Gear C. W. (1971) Numerical initial value problems in ordinary differential equations, Prentice Hall, Englewood Cliffs, New Jersey.

Hairer E. (1984) Solving ordinary differential equations, Springer-Verlag, New York.

Harley R. A., Russell A. G., McRae G. J., Cass G. R. and Seinfeld, J. H. (1993) Photochemical modeling of the Southern California air quality study. *Environ. Sci. Technol.* **27,** 378-388.

Hesstvedt E., Hov Ø., and Isaksen I. S. A. (1978) Quasi-steady-state approximations in air pollution modeling: comparison of two numerical schemes for oxidant prediction. *Int. J. Chem. Kinetics.* **10,** 971-994.

Hindmarsh A. C. (1980) LSODE and LSODI, two new initial value ordinary differential equation solvers *ACM-SIGNUM.* **15-4,** 10-11.

Hov Ø., Zlatev Z., Berkowicz R., Eliassen A. and Prahm L. P. (1989) Comparison of numerical techniques for use in air pollution models with nonlinear chemical reactions. *Atmospheric Environment* **23,** 967-983.

Lapidus L. and Seinfeld J. H (1971) Numerical solution of ordinary differential equations, Academic Press, New York.

Lurmann F. W., Carter W. P. L., and Coyner L. A. (1987) A surrogate species

chemical reaction mechanism for urban-scale air quality simulation models. EPA No. 68-02-4104.

Morris R. E. and Myers T. C. (1990) User's guide for the Urban Airshed Model—Volume I: User's manual for UAM(CB-IV). U.S. Environmental Protection Agency (EPA-450/4-90-007a).

Odman M. T., Kumar N. and Russell A. G. (1992) A comparison of fast chemical kinetic solvers for air-quality modeling. *Atmospheric Environment* **26,** 1783-1789.

Saylor R. D. and Fernandes R. I. (1993) On the parallelization of a comprehensive regional-scale air quality model. *Atmospheric Environment* **27A,** 625-631.

Shin W. C. and Carmichael G. R (1992) Comprehensive air pollution modeling on a multiprocessor system. *Computers chem. Engng.* **16,** 805-815.

Stockwell W. R., Middlenton P., Chang J. S. and Tang, X. (1990) The second generation regional acid deposition model chemical mechanism for regional air quality modeling *J. Geophys. Res.* **95,** 16343-16368.

Stoer J. and Burlisch R. (1980) Introduction to numerical analysis, SpringerVerlag, New York.

Young T. R. and Boris J. P. (1977) A numerical technique for solving stiff ordinary differential equations associated with the chemical kinetics of reactive-flow problems. *J. Phys. Chem.* **81,** 2424-2427.

Chapter 4

Parallel Computation of Atmospheric Chemical Modeling

Donald Dabdub and John H. Seinfeld.

Parallel Computing, In Press.

The porting of atmospheric chemical dynamic models to massively parallel computers presents interesting computational challenges. Strategies for the parallelization of the transport and chemistry operators of atmospheric models are outlined. The use of parallel buffers to perform input/output operations is described. Results are given for implementation of the CIT urban air pollution model on distributed memory multiple instruction / multiple data (MIMD) machines ranging from a cluster of workstations to a 512 node Intel Paragon. A speed-up factor of 94.9 is achieved when the I/O, transport, and chemistry portions of the model are performed in parallel using 128 nodes of the Intel Paragon.

4.1 Introduction

Atmospheric chemical dynamic models are mathematical descriptions of atmospheric transport and transformation that predict how species concentrations change in response to changes in emissions over scales ranging from urban ($\sim 10^4$ km^2) to regional ($\sim 10^6$ km^2) to global. Such models are, for example, the computational tools used to determine emission control strategies needed to comply with urban air quality standards (National Research Council, 1991). At the urban and regional scales, these models are often referred to as air quality models (AQMs).

The basic approach to air quality modeling is Eulerian, in which atmospheric dynamics are simulated on a three-dimensional Cartesian grid. Typical horizontal grid cell dimensions are about 5 km $\times$ 5 km for urban applications and 100 km $\times$ 100 km for regional applications. Current urban-scale atmospheric models subdivide the vertical dimension into 4-7 layers (up to an altitude of $\sim$ 1500m), while regional models use 6-15 layers (up to $\sim$ 10000m). These resolutions are consistent with those of source emission inventories and the ability to resolve meteorological processes that drive the flow fields. Increasing the grid resolution increases the computational cost as well as the data acquisition requirements. Table 4.1 presents some characteristics of existing and anticipated AQMs.

Historically, the degree of physics and chemistry incorporated in atmospheric models has been limited to some extent by the computational power available (Seinfeld, 1989). The basic atmospheric chemical dynamic model describes gas-phase species. Pilinis and Seinfeld (1988) incorporated the particle (aerosol) phase into a three-dimensional gas-phase AQM, leading to an increase in computing time by a factor of 3. Other models (Charmichael *et al.,* 1986; Chang *et al.,* 1987; Venkatram *et al.,* 1988) incorporate aqueous-phase processes in an effort to simulate acid deposition phenomena. As shown in Table 4.1, the next-generation of AQMs is anticipated to incorporate both aqueous and aerosol phases. These next-generation models will require state-of-the-art massively parallel supercomputers to achieve reasonable[1] wall-

[1]Less than 1-hour of computing time per 24-hours of simulation time.

Table 4.1: Aspects of current urban and regional air quality models.

Model	Scale	Vertical Resolution	Horizontal Resolution	Aqueous Phase	Aerosol Phase
UAM Morris and Myers [20]	Urban	4 layers up to 1.5 km	4-10 km	No	No
CIT McRae and Seinfeld [18] Harley *et al.* [13]	Urban	5 layers up to 1.5 km	5 km	No	No
ROM Schere and Wayland [27]	Regional	3 layers up to 2 km	18 km	No	No
RADM-II Chang *et al.* [7]	Regional	15 layers up to 10 km	80 km	Yes	No
STEM-II Carmichael *et al.* [5]	Regional	10 to 14 layers up to 6 km	10-56 km	Yes	No
ADOM Venkatram *et al.* [35]	Regional	12 layers up to 10 km	60-120 km	Yes	No
Next Generation AQMs	Urban and Regional	25-30 layers up to 10 km	4-80 km	Yes	Yes

time turnaround. Clusters of workstations will provide the entry-level computational power demanded by models of such complexity.

The goal of this paper is to address the techniques for parallel computation with atmospheric models by means of implementation of a state-of-the-art urban-scale air quality model on a massively parallel architecture. Table 4.2 summarizes some of the current advances towards the efficient parallel implementation of AQMs. Pai and Tsang (1992; 1993) studied splitting finite element techniques used in the parallel solution of operators in air pollution modeling. Shin and Carmichael (1992) parallelized the chemistry operator of an AQM using shared MIMD architectures. Saylor and Fernandes (1993) and Dabdub and Seinfeld (1994a) parallelized the chemistry and transport operators of AQMs. In the work reported here we: current parallel implementation techniques of chemistry and transport computations; present techniques to parallelize input/output routines; and develop an AQM code independent of the number of nodes available, modular, and portable across distributed MIMD architectures using different message passing libraries.

After the model is described in Section 2, we discuss in Section 3 the techniques used in the parallelization of the chemistry operator of AQMs. Emphasis is placed on the paradigms used. Section 4 studies and compares the performance of different parallel implementations of the atmospheric transport operator. We describe in Section 5 the implementation of buffering techniques used to minimize data read/write operations and the implementation of the input/output operations in parallel. Finally, Section 6 comments on portability issues to achieve machine independence within distributed memory MIMD architectures.

4.2 Model Description

The governing equation of three-dimensional Eulerian gas-phase AQMs is the atmospheric diffusion equation (Seinfeld, 1986):

$$\frac{\partial c_i}{\partial t} + \mathbf{u} \cdot \nabla c_i = \nabla \cdot (\mathbf{K} \cdot \nabla c_i) + R_i(\mathbf{c}, T) + S_i(\mathbf{x}, t) \tag{4.1}$$

Table 4.2: Parallel implementation of urban and regional atmospheric models.

Reference	Work Description	Maximum Speed-up	Architecture
Pai and Tsang [23]	Investigate the effectiveness of parallelization of time-splitting finite element schemes using MIMD machines. Asses the portability of schemes to shared-memory parallel computers.	13.1	Sequent Symmetry S81
Pai and Tsang [24]	Develop a parallel time-splitting finite element scheme for three-dimensional first- and second-order dispersion models to simulate plume behavior of passive contaminants in a convective boundary layer.	5.2	IBM 3090-600J
Shin and Carmichael [31]	Parallel implementation of STEM-II on a shared-memory MIMD machine. Chemistry implemented in parallel. Concluded that vectorization produces inferior performance than concurrency.	2.5	Alliant FX/8
Saylor and Fernandes [26]	Parallel implementation of STEM-II on a shared-memory MIMD machine. Chemistry and transport implemented in parallel.	4.6	IBM 3090-600J
Abramson *et al.* [1]	Parallel implementation of UAM on a shared-memory MIMD machine. Experiments with automatic parallelizing compilers.	6.8	Silicon Graphics SGI 4D/380 S
Dabdub and Seinfeld [8]	Parallel implementation of CIT on a distributed memory MIMD machine. Chemistry and transport implemented in parallel.	28.5	Intel Touchstone Delta
Dabdub and Seinfeld [9]	Parallel implementation of CIT on a distributed memory MIMD machine. Develop new strategies to implement transport operator in parallel.	87.7	Intel Touchstone Delta
This work	Implementation of chemistry, transport, and I/O in parallel. Develop buffering techniques to reduce communication. Develop machine independent code.	94.9	Intel Paragon

where $c_i(\mathbf{x}, t)$ are the elements of the gas-phase species concentration vector $\mathbf{c}$, t is time, $\mathbf{x} = (x, y, z)$, $\mathbf{u} = (u, v, w)$ is the advective flow field, $\mathbf{K}$ is the eddy diffusivity tensor, R_i is the chemical production of species i, T is the temperature, and S_i is the volumetric source of i from emissions (Ground-level emissions are included in the surface-level boundary conditions to Equation (4.1).) The flow field $\mathbf{u} = (\mathbf{x}, t)$ is provided as input to Equation (4.1), either from an interpolation and extrapolation of data or from a solution of the primitive flow equations. We do not address here the problem of meteorological modeling to generate $\mathbf{u} = (\mathbf{x}, t)$.

Atmospheric aerosols are ubiquitous and often observable by eye as dust, smoke, and haze. Particles comprising the atmospheric aerosol range in sizes from nanometers (nm) to tens of micrometers (μm)—that is, from large clusters of molecules to visible flecks of dust. Most of the smallest particles (less than $\sim$ 1 μm) are produced by condensation, either from reactive gases in the atmosphere (e.g., sulfur dioxide) or in high temperature processes (e.g., fire). Particles larger than $\sim$ 1 μm are usually from mechanical production (wind blown soil, sea spray, etc.). As a result of myriad production processes, atmospheric aerosol chemical composition is highly variable, both with size and spatially/temporally. The aerosol is said to be internally mixed when all particles of a given size have the same composition. When the composition is not a unique function of size the aerosol is called externally mixed. The general dynamic equation that describes the evolution of the composition and size of an internally mixed aerosol is (Wexler *et al.*, 1994)

$$\frac{\partial q_i(m, \mathbf{x}, t)}{\partial t} + (\mathbf{u} - V_s(m)\mathbf{k}) \cdot \nabla q_i = H_i(m, \mathbf{x}, t) q(m, \mathbf{x}, t) - \frac{(m q_i H)}{\partial m}$$

$$+ \int_0^m \Gamma(m', m - m', \mathbf{x}, t) q_i(m', \mathbf{x}, t) \frac{q(m - m', \mathbf{x}, t)}{m - m'} dm'$$

$$- q_i(m, \mathbf{x}, t) \int_0^\infty \Gamma(m', m, \mathbf{x}, t) \frac{q(m', \mathbf{x}, t)}{m'} dm' + \nabla \cdot (\mathbf{K}(\mathbf{x}, t) \nabla q_i(m, \mathbf{x}, t))$$

$$+S_i(m,\mathbf{x},t) + R_i(m,\mathbf{x},t) + N_i(m,\mathbf{x},t), \tag{4.2}$$

where $q(m,\mathbf{x},t)$ is the total mass distribution such that $q_i(m,\mathbf{x},t)dm$ is the mass concentration of species i in the mass range $[m, m+dm]$ and $\sum_i q_i = q$, m_i is the mass of species i in a particle of total mass m, H_i is the inverse of the characteristic time for particle growth from condensation or evaporation of species i, $\Gamma(m',m) = \Gamma(m,m')$ is the binary coagulation coefficient, $\mathbf{k}$ is the unit vector in the vertical direction, V_s is the particle gravitational settling velocity, and N_i the nucleation rate of species i in the mass range $[m, m+dm]$. In the present work, while we will confine our attention to gas-phase models, based on Equation (4.1), the general considerations will apply to models that couple gas and particle phases. (Coupling of Equation (4.2) to Equation (4.1) occurs through H_i, R_i, and N_i.)

The direct numerical solution of Equation (4.1) using fully implicit methods leads to systems of linear equations that are prohibitively expensive to solve on most sequential machines. Therefore, AQMs use splitting methods, alternating direction implicit (ADI) techniques, and locally one-dimensional (LOD) methods to solve Equation (4.1). Splitting, ADI, and LOD methods reduce significantly the computational requirements of the models, however they introduce an inherent sequential dependence in the solution of Equation (4.1). When using any of techniques mentioned above, the numerical solution of a particular operator (or dimension) must be solved in sequence. An AQM tailored for parallel computation entails a restructure of the overall numerical techniques, a redesign of the algorithms, and a reorganization of the database containing input data.

This work focuses on the parallel implementation of one particular urban-scale AQM, the CIT model (Harley *et al.,* 1993; McRae *et al.,* 1982a). Other typical urban- and regional-scale AQMs have a structure similar to that of the CIT model; thus there is no loss of generality in considering this model as a test case. Performance data reported here are for the same conditions as those reported in Harley *et al.* (1993) for simulating the air pollution episode on August 27, 1987 in the South Coast Air

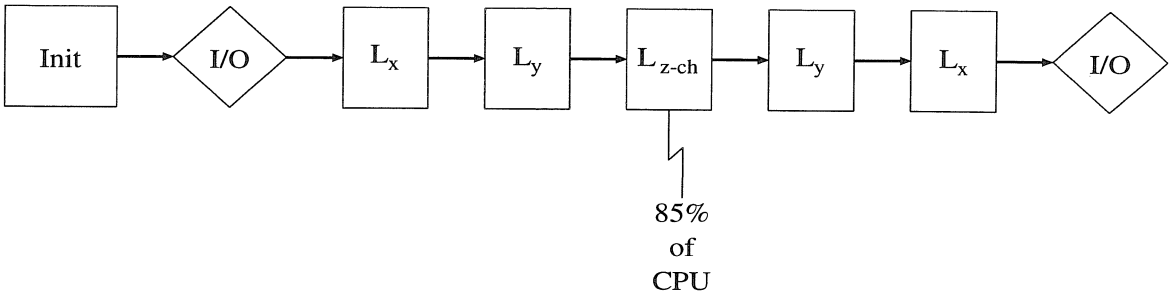


Figure 4.1: Schematic data flow of one time step of sequential implementation of the CIT model.

Basin of California.

The model uses operator splitting techniques to solve Equation (4.1):

$$\mathbf{c}^{t+2\Delta t} = \mathcal{L}_x^{\Delta t}\mathcal{L}_y^{\Delta t}\mathcal{L}_{cz}^{2\Delta t}\mathcal{L}_y^{\Delta t}\mathcal{L}_x^{\Delta t}\mathbf{c}^t, \tag{4.3}$$

where $\mathcal{L}_x$ and $\mathcal{L}_y$ are the advection-diffusion operators in the x and y coordinates, and $\mathcal{L}_{cz}$ is the chemical kinetics and vertical transport operator. Chemical kinetics and vertical transport are coupled into a single operator since these phenomena occur on comparable time scales. Horizontal advection-diffusion is solved using a Galerkin finite-element method (Johnson, 1987; McRae *et al.*, 1982b). The chemistry operator is solved using a hybrid predictor-multi-corrector algorithm (Young and Boris, 1977). Tables 4.3 and 4.4 summarize numerical techniques used in current AQMs to solve the chemistry and advection operators. Figure 4.1 shows the data flow of the sequential model.

Figure 4.2 shows the computational region used to simulate the South Coast Air Basin of California. The grid contains 996 columns; each column consists of 5 grid cells in the vertical direction. Each grid cell contains the concentrations of 35 chemical species together with meteorological data (temperature, relative humidity, solar radiation, and advective flow field). The chemical reaction mechanism includes 106 reactions among the 35 species. These numbers are quite representative of other urban-scale AQMs.

Table 4.3: ODE integration methods for chemical kinetics of Air Quality Models.

Method	Reference	Characteristics
LSODE	Hindmarsh (1980)	General stiff and nonstiff solver. Uses Adams methods (predictor-corrector) in the nonstiff case, and backward differentiation methods for the stiff case. Treats Jacobian matrix as a full or banded matrix. The linear systems that arise are solved by direct methods (LU factor/solve).
VODPK	Brown and Hindmarsh (1989)	General stiff and nonstiff solver. Uses Adams methods in the nonstiff case, and backward differentiation methods for the stiff case. Uses Krylov subspace iterative methods with right, left or no preconditioners and scaling.
QSSA	Hesstvedt, *et al.* (1978)	Stiff solver specifically designed for chemical kinetics problems. Uses analytical characteristics of rate equations for integration. Fixed time step. Some stability problems in long time simulations.
Hybrid	Young and Boris (1977)	Stiff solver specifically designed for chemical kinetics problems. Uses asymptotic approximations to rate equations for integration. Variable time step. Predictor-multi-corrector algorithm.
Extrapolation	Dabdub and Seinfeld (In press)	A general solver that uses Richardson extrapolation to improve accuracy over single time step solution. To solve chemical kinetics problems the algorithm proposed in this work uses a modified QSSA scheme.

Table 4.4: Numerical advection methods used in Air Quality Models.

Method	Characteristics	Reference
SMOL Smolarkiewicz Method	Iterative scheme based on upstream differences. Positive definite.	Smolarkiewicz (1984)
NMOL Numerical Method of lines	Fourth-order directional difference in space. SDRIV2 integration in time. Produces negative conc.	Schiesser (1991) Carver and Hinds (1978) Kahaner *et al.* (1989)
BOTT Bott Method	Nonlinear renormalization of advective fluxes. Positive definite	Bott (1989)
EMDE Emde Method	Continuous curvature cubic-spline. Positive definite.	Emde (1992)
GLRK Galerkin Method	Chapeau function Galerkin finite element. Produces negative conc.	McRae *et al.* (1982b)
ASD Accurate Space Derivative Method	Fourier techniques to accurately compute space derivatives. Third-order Taylor expansion to integrate in time. Produces negative conc.	Gazdag (1973)

We will proceed to study the parallelization of the model based on numerical techniques used in the sequential code. The task of designing an *ab initio* tailored numerical scheme to solve the atmospheric diffusion equation on a parallel architecture is left to further research. The initial challenge is the parallelization of the chemistry and transport operators, the core of AQMs. Domain decomposition strategy and coarseness of the parallelization are identified in this step. The second phase of the design of the parallel code is to optimize parallel I/O operations. This aspect has been neglected in previous development of parallel AQMs.

4.3 Parallelization of the Chemistry Operator

Approximately 85% of the CPU time in a typical urban or regional air pollution simulation is spent in the solution of the chemistry operator. Since the data required to solve the chemistry operator are local for each grid column, this suggests a starting point towards the parallelization of the model using a single producer/many consumers paradigm. The producer's (host's) task is to read and pack the data required

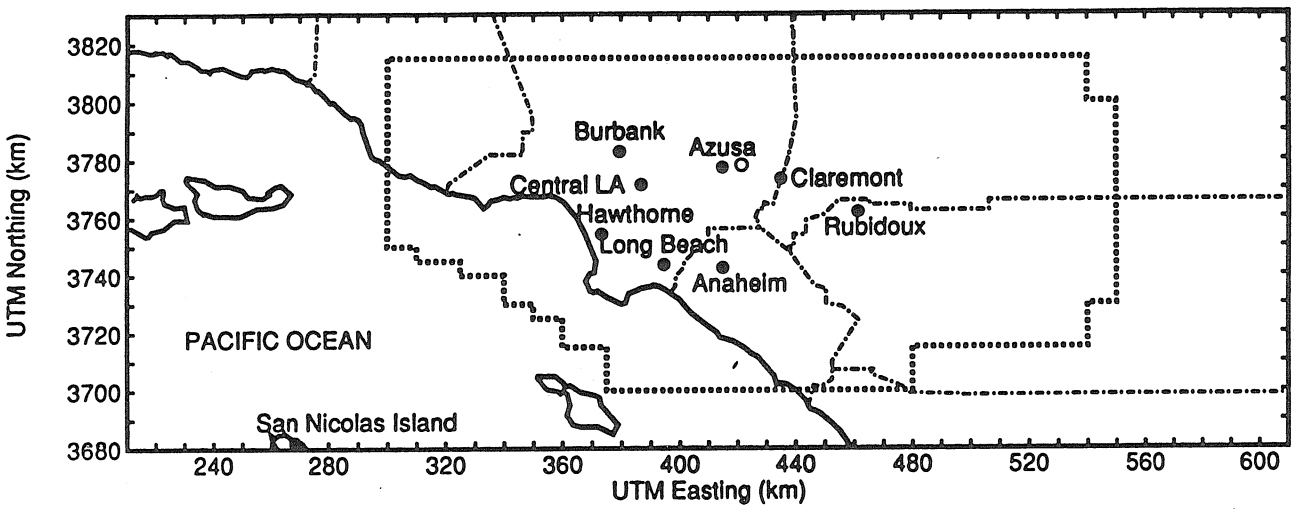


Figure 4.2: Modeling region used in simulation of the South Coast Air Basin. The rectangle represents the x-y projection of the master data storage grid of the model. Area enclosed by the dotted line represents the actual computational domain.

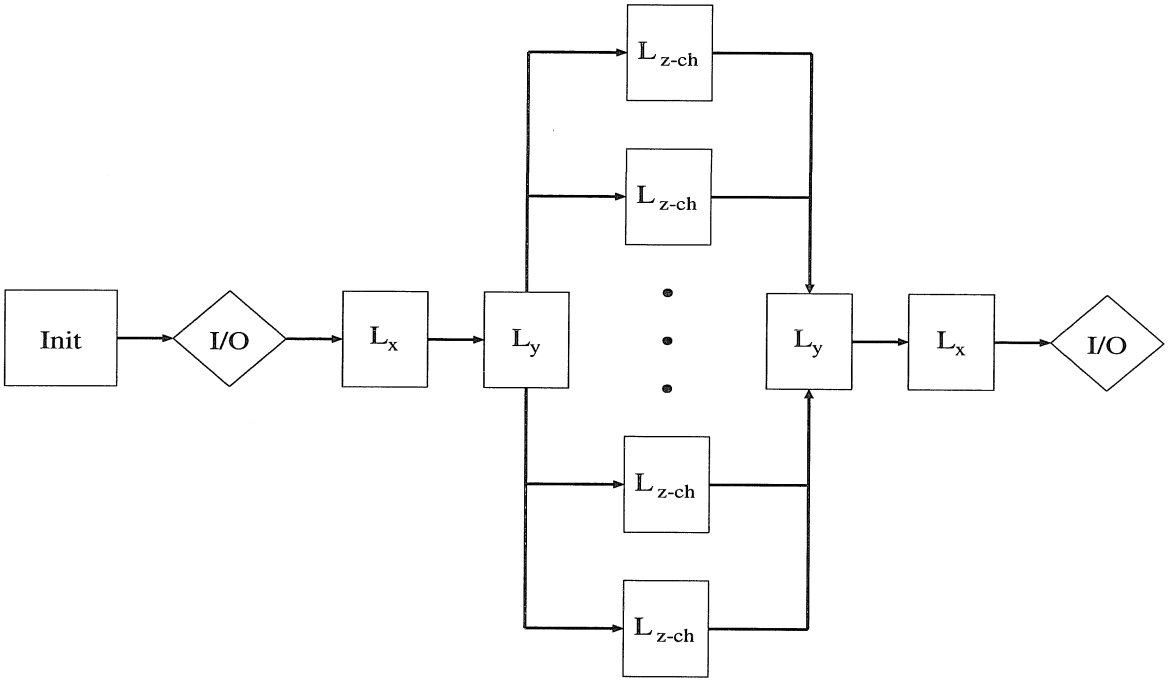


Figure 4.3: Schematic data flow of one time step of the CIT model implementing a producer/consumer paradigm to solve the chemistry operator in parallel.

to solve the chemistry operator, and to perform the horizontal transport computations. The producer also receives the processed data and produces formatted output. Each consumer's (slave's) task is to receive that data, perform the time step integration, and send them back to the producer. Figure 4.3 shows a data flow of this type of parallelism. Note that this approach does not require any communication among consumers.

Parallelization of the chemistry presents the following question: What is the optimal distribution of columns among the nodes? If we assume that the time required to process each column is constant for each time step, the answer is simple; each node takes approximately the same number of columns to process. In each time step a particular consumer node takes the same section of a predetermined domain decomposition. Under this assumption, the performance of the parallel code is independent of the geometry of decomposition implemented.

The assumption of constant time per column must be discussed further. The processing of each column consists of the integration of a system of coupled, stiff,

nonlinear ordinary differential equations representing the chemistry. Processing time may differ from column to column depending on the degree of stiffness of the equations in different areas of the spatial domain. The columns in which a higher degree of stiffness exists require smaller time steps to advance the solution and thus greater overall computation time. If the distribution is one column per consumer, then the load will be slightly off-balanced. In such a situation, the time to solve the chemistry operator each time step is determined by the processing time of the stiffest column. Therefore, one expects the speed-up curves to flatten at the high number of processors regime. When using a low to medium number of processors (compared with the number of columns), the load balance is significantly improved, because of reduced differences in processing time among the consumer nodes.

Two types of variables arise naturally when implementing the vertical transport and chemistry in parallel: constant (those variables that do not change over the course of the simulation, e.g., surface roughness) and variable data (e.g., temperature and relative humidity). Constant data are broadcast to all nodes at the end of the initialization stage. If a predetermined domain decomposition is used, some of the constant data are decomposed before being broadcast. Variable data are stored in a buffer so that, to minimize communication overhead, only one message is passed from producer to consumer.

A predetermined domain decomposition was used by Dabdub and Seinfeld (1994a) on the CIT model and a speed-up of 13.9 was reported when performing only the chemistry in parallel. The use of a predetermined decomposition has two main advantages. First, it simplifies the debugging process; the programmer knows which columns are being sent to which nodes. Second, it reduces the communication of sending initial data that can be decomposed. For example, if a determined decomposition is used the entire topographical data need not be sent to all the nodes. Instead, only the section of the predetermined domain is sent from the host.

Figure 4.9, to be shown later, presents the measured times for a standard 24-hour simulation of the California South Coast Air Basin using the CIT model after parallelizing the chemistry operator only. The simulation was performed on the Intel

Touchstone Delta using NX as the message passing library. The speed-up curve is monotonically decreasing. However, the rate of change of the curve diminishes (almost flattens) in the high number of processors regime.

Parallel implementation of the chemistry, as described above, is a straightforward approach, requiring only introductory parallel programming techniques. Furthermore, because of its simple data flow structure, a predetermined decomposition avoids the most common pitfalls of MIMD programming: algorithmic and cyclic deadlocks as well as process starvation. One of the major problems of the single producer/many consumers paradigm is that the producer might not be distributing the data sufficiently fast for all the consumers. This effect explains the flattening of the chemistry curve.

4.4 Parallelization of the Horizontal Advection Operator

The advection equation, one of the component operators in the splitting scheme, accounts for the horizontal transport of pollutants under a given wind field. In the sequential mode the advection solution requires only about 5% of the CPU time when a Galerkin method is used. Whereas this is a small percentage of the overall sequential time, it represents a significant percentage of the CPU time once the chemistry has been parallelized. Therefore, parallelizing the transport operator promises higher speed-up than that obtained by parallelizing chemistry alone. The basic idea is to perform the computations involved in the solution of the advection equation of all rows (or columns) simultaneously. Since the data required to advance the solution are distributed among different nodes, there is an inherent need for extensive interprocessor communication. Thus, parallelizing the transport operator is more challenging than the chemistry operator. Furthermore, different advection solvers require different amounts of data. To maintain the modular design of the code we send all possible data that might be required to the node performing the transport

computations. The extra communication costs provide freedom to use any advection solver and an infrastructure to solve the transport operator that is independent of the domain decomposition used by the solution of the chemistry operator.

The general approach for implementing the transport is to use the foreman-worker paradigm (See Figure 4.4). The foreman (host node) divides the computational work among the workers (slave nodes). The workers are arranged in such an order that they can communicate with the foreman and with any other worker. Given this configuration, some questions need to be answered: How will the work be distributed? After each transport process has modified its data set, it must return the new concentrations to its node of origin. Then: Which node is the originating one? Is the originating node ready to receive the modified data? From the originating node perspective, after the data have been modified in different places, how does it know which data are coming from which node? Could there be a situation where two workers are attempting to send data to each other? Does the communication strategy guarantee a one-to-one correspondence between the sends and receives? Finally, will the communication strategy still be valid if the numerical solution to the transport operator is changed?

Different answers to these questions represent different strategies. Dabdub and Seinfeld (1994b) selected one of the extreme nodes of each row (or column) as the worker that will process the transport of all species of the column for all layers. A buffer is created for each species and layer containing the required data to solve the transport operator. The buffer is appended information describing the origin and destination worker of each species. To avoid deadlock situations, algorithm synchronization is implemented as the first step of the transport operator. This approach is known as Designated Transport Node (DTN) (Dabdub and Seinfeld, 1994b). Figure 4.5 shows the data distribution used by DTN when solving the advection-diffusion operator in the $x-$direction. Using DTN a speed-up of about 30 with chemistry in parallel is achieved as will be presented subsequently in Figure 4.9. Results show that in the low number of processors regime, the overhead in message communication used in the transport implementation overshadows the reduced amount of work performed

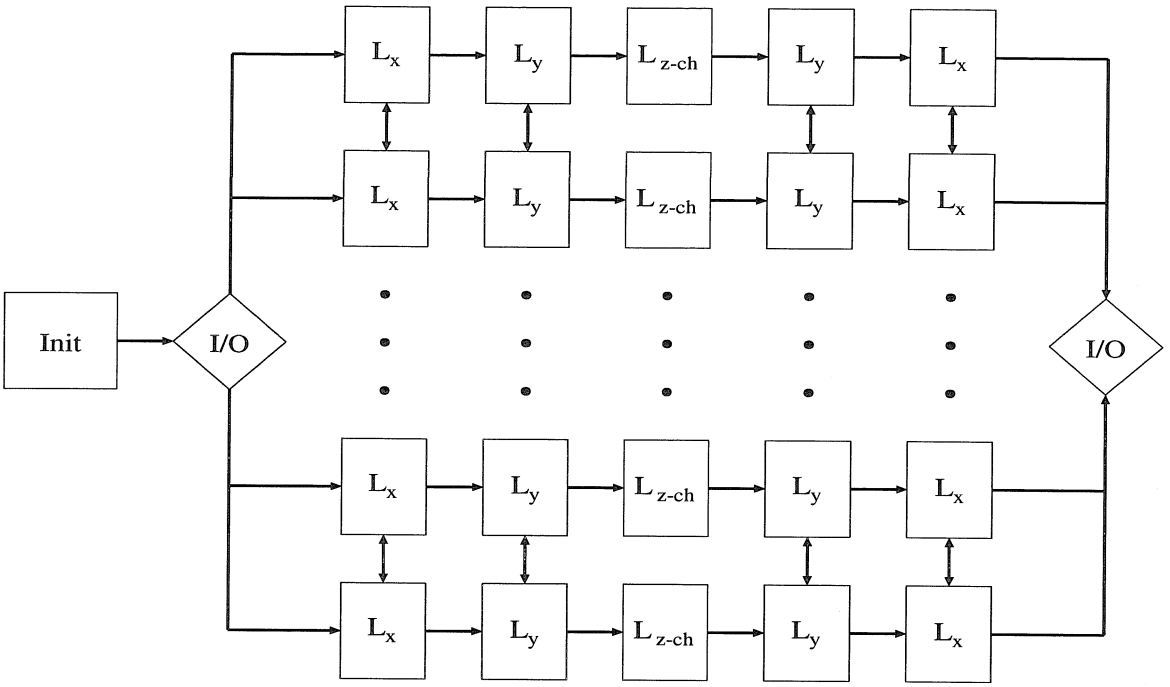


Figure 4.4: Schematic data flow of one time step of the CIT model implementing a foreman/worker paradigm to solve the transport operator in parallel.

by the host. The DTN approach has the advantage that the underlying message passing structure is still valid if the algorithm to solve the advection equation is modified. On the other hand, one problem with DTN is that when the number of workers is greater than the number of rows (or columns) the overall efficiency is decreased. In such a situation the extra workers will be idle while the transport computations are being carried out.

To overcome the idle time of the extra workers when the number of workers is greater than the number of rows, a dynamical decomposition approach is implemented. The Dynamical Balanced (DYB) (Dabdub and Seinfeld, 1994b) approach consists of distributing the first N rows of the lower layer to the first N workers. N is the number of available workers. That is, instead of processing all the species for all the layers in a single worker, the different layers are distributed among all other nodes. The DYB approach uses the same foreman-worker paradigm illustrated in Figure 4.4, but with a higher number of workers solving the $\mathcal{L}_x$ and $\mathcal{L}_y$ operators. Figure 4.6 shows the data distribution used by DYB when solving the advection-diffusion

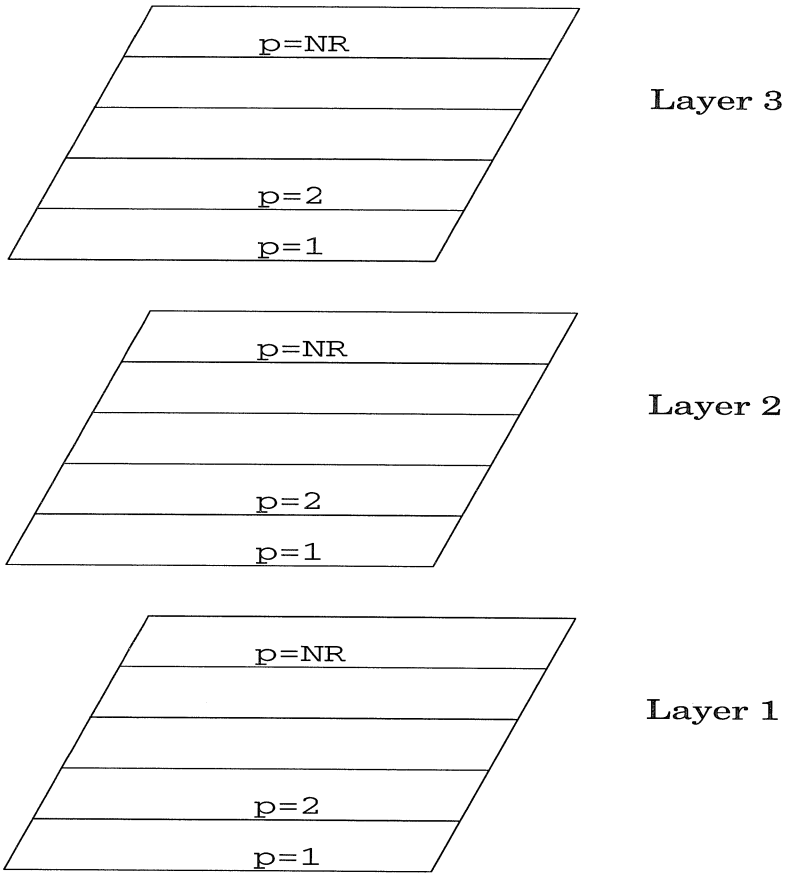


Figure 4.5: Example of row distribution to solve the advection- diffusion operator in the $x-$direction using a DTN approach. Each row contains the processor number where the advection computations for such row are to be performed. The maximum number of processors used in the solution of the operator is number of rows (NR) of the domain.

operator in the $x-$direction. The DYB approach requires a more elaborate use of indexing and pointers in the code. In addition, since all the nodes perform transport computations, the required transport data need to be sent to all the nodes. As long as the time used by the extra communication is smaller than the time saved by the extra workers, the DYB approach is worth implementing. As it will be shown subsequently in Figure 4.9, there is no advantage of using the DYB approach when a fast transport solver such as the Galerkin method is used. On the other hand, when using the more computationally intensive Accurate Space Derivative method (ASD), the advantages of using the DYB strategy are clear, as will be shown in Figure 4.10. The ASD technique, a highly accurate algorithm to solve the advection operator (Gazdag, 1973), imposes such a high computational demand that it is almost prohibitively expensive to use on sequential computers. The fact that ASD has a high computational requirement makes its computation to communication ratio greater than other techniques. As a result, it is more attractive for use in a parallel environment. Parallel computers not only permit the use of a highly demanding algorithm, but also encourage its use.

4.5 Parallelization of the I/O

Up to this point we have discussed parallelization of the chemistry and the advection operators of AQMs. Can anything more be done at this point to further increase the speed-up of the model? It is observed from Figure 4.4 that all nodes must wait idly while the host is reading and writing data to the file system. The I/O is not a major fraction of the total time of the code when executed sequentially (about 5%). However, at this point when the two main operators of the model have been parallelized, the I/O becomes a significant bottleneck that affects all the worker nodes. There are several approaches to overcome the I/O bottleneck that are discussed in this section.

The most alluring strategy seems to eliminate the I/O from the host node. In that case, the bottleneck would completely disappear. Furthermore, there would not be any messages that need to be sent. In this strategy all workers perform their own I/O.

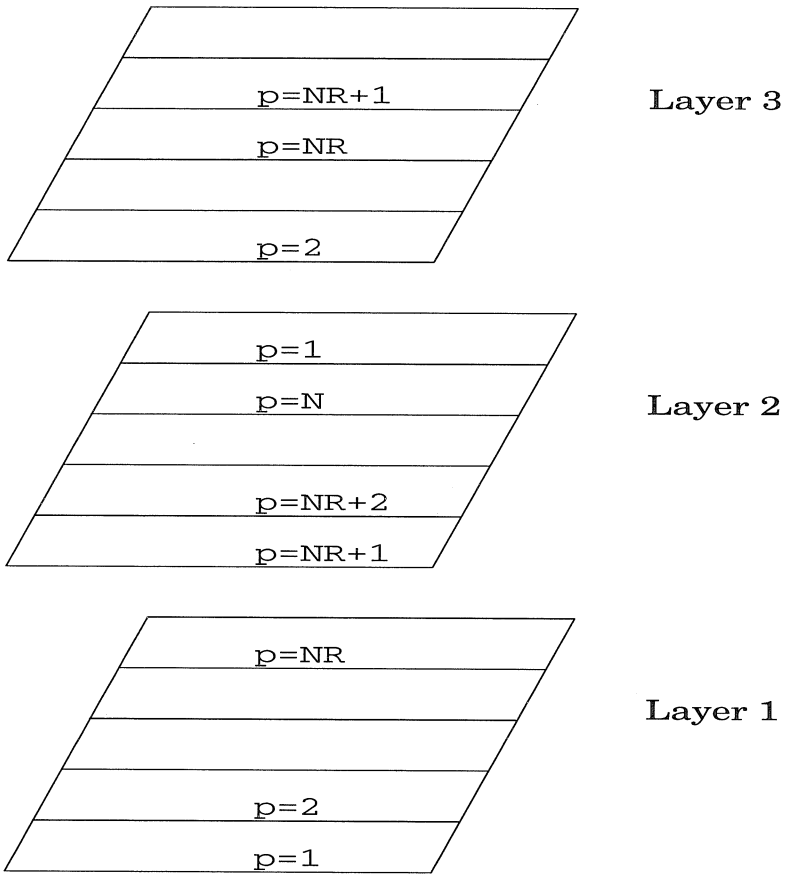


Figure 4.6: Example of row distribution to solve the advection- diffusion operator in the $x-$direction using a DYB approach. Each row contains the processor number where the advection computations for such row are to be performed. All N processors are involved in the advection-diffusion computations by decoupling the vertical layers to achieve higher parallelism. The number of rows (NR) of the domain is not a limiting factor on the number of processors used.

Despite its promising potential, this idea turns out to be disastrous. When there is a high number of computational nodes trying to access the file system, the performance of the I/O operations is hindered by the operating system overhead. Worse yet, when the number of nodes is sufficiently large, the entire file system collapses. Another strategy must be considered.

When all the workers are performing chemistry and transport computations the host is waiting idly for the results to arrive. Once the results arrive they are organized, formatted, and written to the output files. The use of this idle time is the basis of the next step to increase the performance of the code. To accomplish that, a first-in-first-out (FIFO) queue is created. When the host reads the data initially it sends them to the worker nodes. Instead of waiting for the workers to send the processed data back, the host proceeds immediately to read the input data for the next time step. The host decomposes and places the newly read data in the input queue while the workers are still performing the chemistry and transport computations. The use of the queue consumes the same amount of message passing time but reduces the reading time. The overhead of keeping track of the queue is small compared to that saved during the reading process. The data flow is illustrated in Figure 4.7.

Performance measurements show that in the high number of processors regime the host is still a bottleneck, even after eliminating its idle time. The workers are still processing the data faster than the foreman is able to read and distribute them. The next step taken is to eliminate one of the workers and to create a second host. That is, the paradigm now used is that of two foremen and many workers, in which each of the foremen performs approximately half the task of the original foreman. The new data flow is illustrated in Figure 4.8. The two foremen read different files simultaneously. Two processes reading data do not present any problems in the file system performance. Another benefit of using two foremen is that the message passing time is reduced by almost half. Therefore, the parallelization of the I/O described here focuses on distributing file access operations between two hosts. In addition, the Intel Paragon I/O sub-system is automatically distributing or striping the data across the disks.

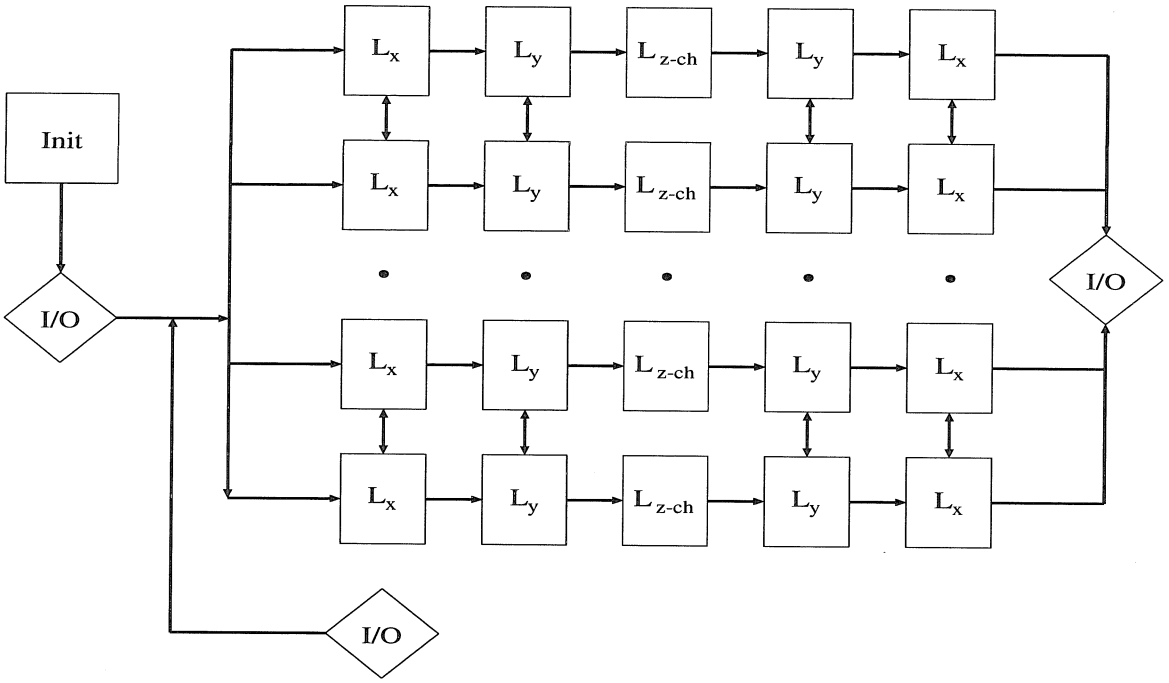


Figure 4.7: Schematic data flow of one time step of the CIT model implementing a buffering queue for input/output operations.

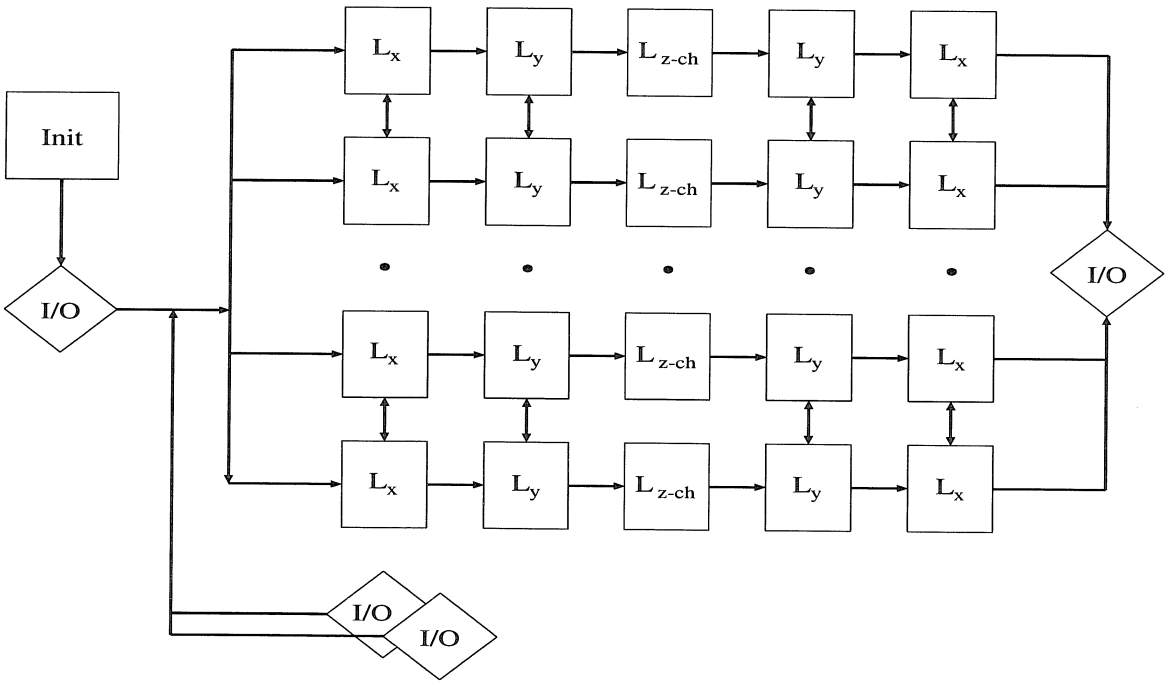


Figure 4.8: Schematic data flow of one time step of the CIT model implementing parallel buffering queue for input/output operations.

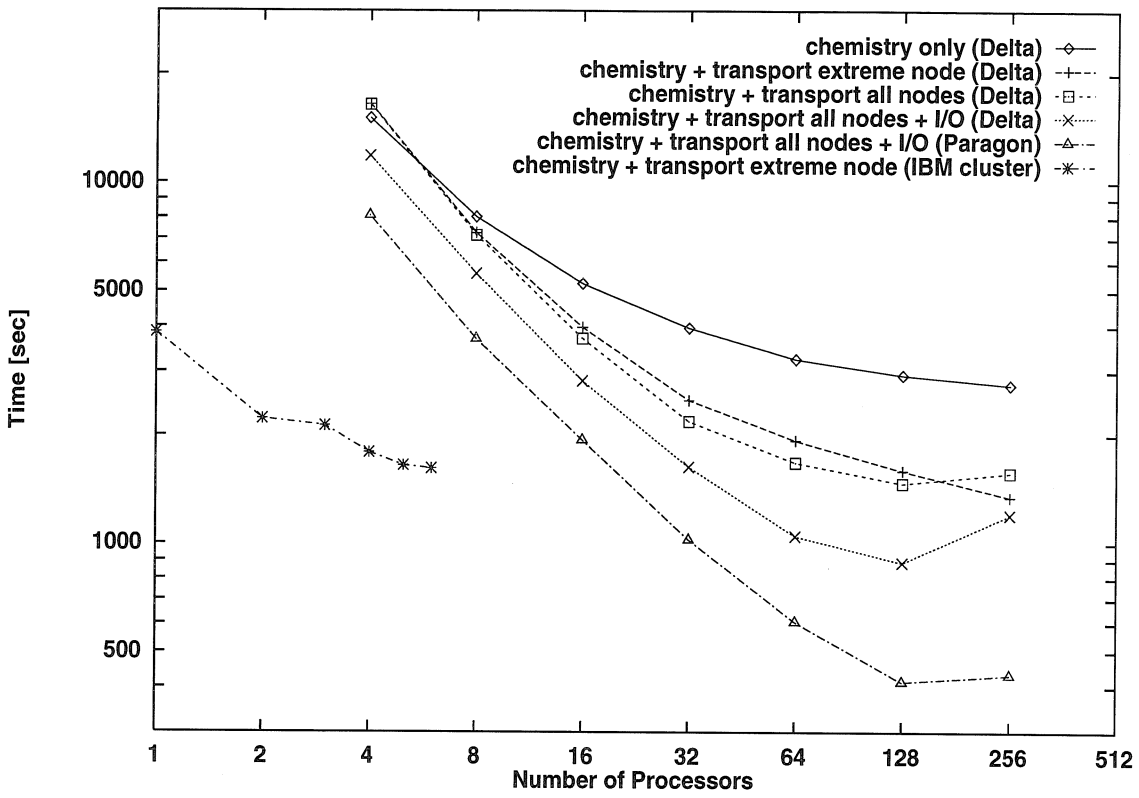


Figure 4.9: CPU time for a 24-hour simulation of the South Coast Air Basin versus number of nodes for the Galerkin transport implementation of the CIT using different parallelization strategies.

Figure 4.9 shows that the performance of the model significantly increases after the I/O has been implemented using the queue and two foremen technique. Figure 4.10 also shows a dramatic improvement when using the more computationally intensive ASD algorithm to solve the transport operator. The fastest run measured using the Galerkin solver is on the Intel Paragon with 128 nodes. A 24-hour simulation consumes less than 7 minutes which corresponds to a speed-up of about 45. The fastest run measured using the ASD solver is also on the Intel Paragon with 128 nodes. A 24-hour simulation requires 11.5 minutes, which corresponds to a speed-up factor of 94.

When the model is highly parallelized there is a performance decrease when adding extra processors in the high number of processors regime. This effect occurs because

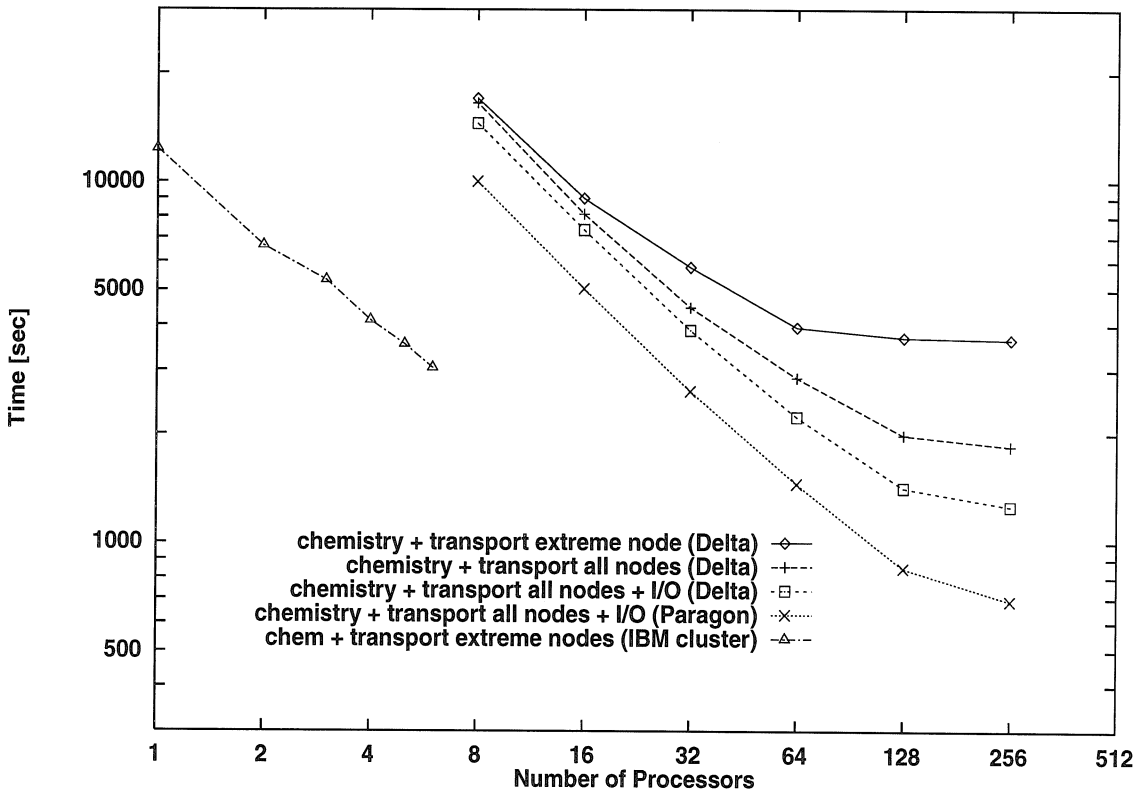


Figure 4.10: CPU time for a 24-hour simulation of the South Coast Air Basin versus number of nodes for the ASD transport implementation of the CIT model using different parallelization strategies.

the computation to communication ratio inherent in the model is not sufficiently high. Stated in a different manner, the task of simulating this Los Angeles Basin gas-phase air pollution episode is sufficiently large to run optimally on 128 nodes but not on 256 or more processors. Moreover, it can be seen that the speed-up of the model using the ASD method is higher than that obtained using the Galerkin method since the former has a higher computation to communication ratio.

4.6 Key Issues on Portability

The use of the NX message passing library provides various parallel computers commercially available as platforms to run the model: the Touchstone Delta, the Intel Paragon, the Gamma machine, the IPSC and the IPSC2. However, a few workstations connected in a network are more likely to be accessible than such machines. Moreover, there is a wide variety of distributed MIMD type architecture that do not support the NX library. It is crucial for the wide assessibiltity of parallel air quality models that they be portable across different computing platforms.

With that motivation, we first implemented a parallel version of the model using P4 as the message passing library (Butler and Lusk, 1994). The cluster of workstations is considered a MIMD distributed memory concurrent computer. Under that perspective the porting of the NX model to the P4 environment consists of modifying the calls and syntax of the communication library. A script that automates such conversion was designed. Processor efficiencies of 50-80% are obtained when running the model with the Galerkin advection solver on a cluster of workstations. When the ASD solver is used processor efficiencies reach 80-90% because of the higher computation to communication ratio inherent to ASD. The cluster is made up of IBM RISC 390 nodes and a IBM RISC 580 host using a CDDI network. The scalability of the model with the Galerkin transport solver on the network of workstations is hindered by several factors. First, the Intel Delta and Paragon machines have significantly higher network bandwidth as shown in Figure 4.11. In addition, the performance of CDDI and Ethernet networks degrades linearly with the number of processors. Third,

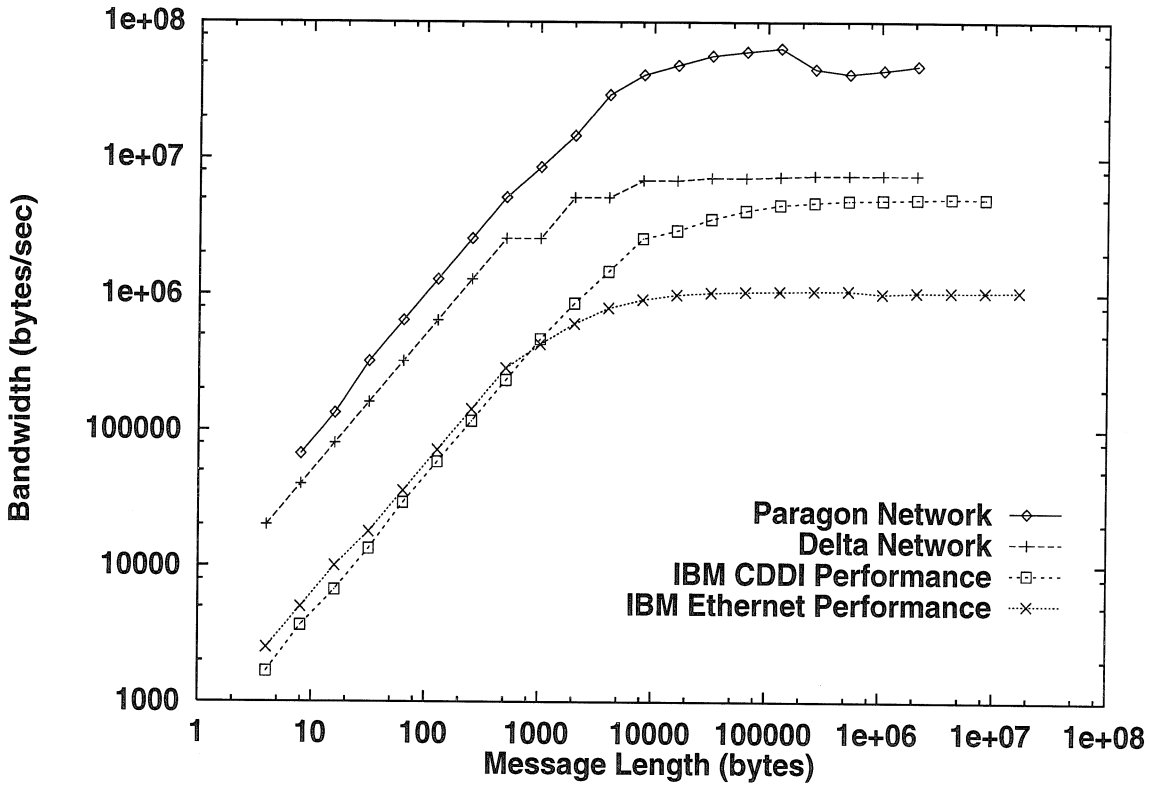


Figure 4.11: Message passing performance of various networks used. Bandwidth measurements correspond to rendez-vous sends.

the performance of the compute nodes of the network of workstations (RS/6000-390 processors) is significantly greater than those of the Delta and Paragon machine (Intel i860 processors). Figure 4.10 shows that increasing the computation per node, by using ASD, improves the scalability of the network of workstations.

Regardless of the availability of the script, it is still cumbersome to manage software development when different codes for different environments are available. This need has been the driving force of MPI, a proposed standard message passing interface (Walker, 1994). MPI would permit the point-to-point and global routines under different communication contexts and network topologies. At the time we started developing the parallel code a full implementation of MPI was not available. To overcome the portability of the code we designed our own message passing interface and incorporated it into the parallel model. The routines are designed to run on top of P4

and NX. Other libraries such as PVM or EXPRESS can be easily incorporated. The use of such routines produces a "universal" code that runs on different architectures under different basis communication libraries.

4.7 Conclusions

This paper describes the strategies used to implement a typical urban- or regional-scale air quality model in a parallel environment. Parallel implementation of the chemistry operator, transport operator, and I/O routines are required to obtain the highest speed-ups. Using a Galerkin horizontal transport solver a typical model run requires less than 7 minutes on the Intel Paragon with 128 nodes. This corresponds to a speed-up of 45. Using the more computationally intensive ASD transport solver, the same run requires less than 11.5 minutes, which corresponds to a speed-up of 95. The speed-up numbers reported here represent the parallel implementation of a model that was designed to run on a sequential machine. Parallel computing motivates research to find new solution strategies to the atmospheric diffusion equation (with nonlinear chemistry) not envisioned when only uniprocessors are used.

Parallel processing gives the computational power necessary to increase the level of physical and chemical detail in the models. For instance, the incorporation of the aerosol and aqueous phases are now receiving attention by the research community. Aerosol particles play a significant role in human health and urban visibility. The incorporation of the aqueous phase permits the study of acid deposition and acidic fogs. In either case, the addition of new physics will substantially increase the computational task required by the new models. From the data flow perspective, both of these phenomena would require a producer/consumers paradigm similar to that used here to parallelize the chemistry. Therefore, some of the computational "infrastructure" to incorporate new physico-chemical phenomena in parallel has been laid out here. Adding new phenomena to the models increases their computation to communication ratio. Consequently, higher speed-ups are to be expected. A higher degree of complexity in atmospheric chemical dynamic models than currently exists is well

suited for the 500+ number of processors regime.

References

Abramson D., Cope M. and McKenzie R. (1994) Modelling photochemical pollution using parallel distributed computing platforms, in: Halatsis C., Maritsas D., Philokyprou G. and Theodoridis S. *Lecture Notes in Computer Science 817* Springer, New York, U.S.A.

Bott A. (1989) A positive definite advection scheme obtained by nonlinear renormalization of the advective fluxes. *Mon. Wea. Rev.* **117,** 1006-1015.

Brown P. N. and Hindmarsh A. C. (1989) Reduced storage matrix methods in stiff ODE systems. *J. Appl. Math. Comput.* **31,** 40-91.

Butler R. M. and Lusk E. L. (1994) Monitors, messages, and clusters—the p4 parallel programming system. *Parallel Computing* **20,** 547-564.

Carmichael G. R., Peters L. K. and Kitada T. (1986) A second generation model for regional-scale transport/chemistry/deposition. *Atmospheric Environment* **20,** 173-188.

Carver M. B. and Hinds H. W. (1978) The method of lines and the advection equation. *Simulation* **31,** 59-69.

Chang J. S., Brost R. A., Isaksen I. S. A., Madronich S., Middleton P., Stockwell W. R. and Walcek C. J. (1987) A three-dimensional Eulerian acid deposition model: physical concepts and formulation. *J. Geophys. Res.* **92,** 14681-14700.

Dabdub D. and Seinfeld J. H. (1994a) Air quality modeling on Massively Parallel Computers *Atmospheric Environment* **28,** 1679-1687.

Dabdub D. and Seinfeld J. H. (1994b) Numerical advective schemes used in air quality models—sequential and parallel implementation. *Atmospheric Environment* **28,** 3369-3385.

Dabdub D. and Seinfeld J. H. (1995) Extrapolation techniques used in the solution of stiff ODEs associated with chemical kinetics of air quality models. *Atmospheric Environment* **29,** 403-410.

Emde D. V. D. (1992) Solving conservation laws with parabolic and cubic splines *Mon. Wea. Rev.* **120,** 82-92.

Gazdag J. (1973) Numerical convective schemes based on accurate computation of space derivatives. *J. comp. Phys.* **13,** 100-113.

Harley R. A., Russell A. G., McRae G. J., Cass G. R. and Seinfeld J. H. (1993) Photochemical modeling of the Southern California air quality study. *Environ. Sci. Technol.* **27,** 378-388.

Hesstvedt E., Hov Ø., and Isaksen I. S. A. (1978) Quasi-steady-state approximations in air pollution modeling: comparison of two numerical schemes for oxidant prediction. *Int. J. Chem. Kinetics* **10,** 971-994

Hindmarsh A. C. (1980) LSODE and LSODI, two new initial value ordinary differential equation solvers *ACM-SIGNUM* **15,** 10-11.

Kahaner D., Moler C. and Nash S. (1989) *Numerical Methods and Software.* Prentice-Hall Inc., Englewood Cliffs, New Jersey.

Johnson C. (1987) *Numerical Solution of Partial Differential Equations by the Finite Element Method.* Cambridge University Press, England.

McRae G. J., Goodin W. R. and Seinfeld J. H. (1982a) Development of a second generation mathematical model for urban air pollution—I. Model formulation. *Atmospheric Environment* **16,** 679-696.

McRae G. J., Goodin W. R. and Seinfeld J. H. (1982b) Numerical solution of the atmospheric diffusion equation for chemically reactive flows. *J. comp. Phys.* **45,** 1-42.

Morris R. E. and Myers T. C. (1990) User's guide for the Urban Airshed Model—Volume I: User's manual for UAM (CB-IV). U.S. Environmental Protection Agency (EPA-450/4-90-007a).

National Research Council (1991) *Rethinking the ozone problem in urban and regional air pollution.* National Academy Press, Washington, D. C.

Odman M. T., Kumar N. and Russell A. G. (1992) A comparison of fast chemical kinetic solvers for air-quality modeling. *Atmospheric Environment* **26,** 1783-1789.

Pai P. and Tsang T. T. H. (1992) Parallel computations of turbulent-diffusion in convective boundary-layers on shared-memory machines. *Atmospheric Environment* **26A,** 2425-2435.

Pai P. and Tsang T. T. H. (1993) On parallelization of time-dependent 3-dimensional transport-equations in air-pollution modeling. *Atmospheric Environment* **27,** 2009-2015.

Pilinis C. and Seinfeld J. H. (1988) Development and evaluation of an Eulerian photochemical gas-aerosol model. *Atmospheric Environment* **22,** 1985-2001.

Saylor R. D. and Fernandes R. I. (1993) On the parallelization of a comprehensive regional-scale air quality model. *Atmospheric Environment* **27A,** 625-631.

Schere K. L. and Wayland P. K. (1991) EPA regional oxidant model (ROM 2.0): Evaluation on 1980 NEROS databases. U. S. Environmental Protection Agency, Research Triangle Park, N. C.

Schiesser W. E. (1991) *The Numerical Method of Lines.* Academic Press, San Diego, California.

Seinfeld J. H. (1986) *Atmospheric Chemistry and Physics of Air Pollution.* Wiley, New York.

Seinfeld J. H. (1989) Urban air pollution: state of the science. *Science* **243,** 745-752.

Shin W. C. and Carmichael G. R. (1992) Comprehensive air pollution modeling on a multiprocessor system. *Computers Chem. Engng.* **16,** 805-815.

Smolarkiewicz P. K. (1984) A simple positive definite advection scheme with small implicit diffusion. *Mon. Wea. Rev.* **111,** 479-486.

Walker D. W. (1994) The design of a standard message-passing interface for distributed-memory concurrent computers. *Parallel Computing* **20,** 657-673.

Wexler A. S., Lurmann F. W. and Seinfeld J. H. (1994) Modelling urban and regional aerosol—I. Model development *Atmospheric Environment* **28,** 531-546.

Venkatram A., Karamchandani P. K. and Misra P. K. (1988) Testing a comprehensive acid deposition model. *Atmospheric Environment* **22,** 737-747.

Young T. R. and Boris J. P. (1977) A numerical technique for solving stiff ordinary differential equations associated with the chemical kinetics of reactive-flow problems. *J. Phys. Chem.* **81,** 2424-2427.

Chapter 5

Diagnosis of Air Pollution in the San Joaquin Valley of California

5.1 Model Description

The SARMAP model is designed to compute the concentration of atmospheric trace species based on the numerical solution of the atmospheric diffusion equation in an Eulerian modeling framework (Tanrikulu *et al., 1996).* Mathematically, the dynamics of atmospheric pollutants for each species i are described by the following system of conservation equations:

$$\frac{\partial c_i}{\partial t} + \mathbf{u} \cdot \nabla c_i = \nabla \cdot (\mathbf{K} \cdot \nabla c_i) + R_i(\mathbf{c}, T) + S_i(\mathbf{x}, t) + \frac{\partial c_i}{\partial t}|_{clouds}, \tag{5.1}$$

where $c_i(\mathbf{x}, t)$ are the elements of the gas-phase species concentration vector $\mathbf{c}$, t is time, $\mathbf{x} = (x, y, z)$, $\mathbf{u} = (u, v, w)$ is the advective flow field, $\mathbf{K}$ is the eddy diffusivity tensor, R_i is the chemical production of species i, T is the temperature, and S_i is the volumetric source of i from emissions and deposition. The last term in Equation (5.1) accounts for cloud effect processes as described below.

Vertically, SARMAP uses a 15-layer terrain following $\sigma-$coordinate system. All concentrations and meteorological data are defined at the center of each cell. Eddy diffusivities and the vertical component of the advective flow are defined at the boundaries between layers. The $\sigma-$coordinate system used is defined as follows:

$$\sigma = (P - P_{top})/P^{\star} \tag{5.2}$$

$$P^{\star} = P_{surf} - P_{top}, \tag{5.3}$$

where P is the pressure at the level where σ is evaluated, P_{surf} the surface pressure, P_{top} the pressure at the top of the modeling region (16000 m). Table 5.1 shows the discretization of the grid in the vertical direction.

The initial conditions used when solving Equation (5.1) are obtained from available measurements of vertical profiles $c_{(i,0)}(z)$

$$c_i(\mathbf{x}, t) = c_{(i,0)}(z) \tag{5.4}$$

Table 5.1: Discretization of the grid using σ-coordinates.

Level Index	σ-Index	Standard Pressure (kPa)	Standard Height (m)
15	0.0	10.0	16069
14	0.1	19.0	11998
13	0.2	28.0	9512
12	0.3	37.0	7621
11	0.4	46.0	6073
10	0.5	55.0	4754
9	0.6	64.0	3600
8	0.7	73.0	2570
7	0.78	80.2	1818
6	0.84	85.6	1289
5	0.89	90.1	868
4	0.93	93.7	544
3	0.96	96.4	307
2	0.98	98.2	152
1	0.99	99.1	76
0	1.00	100.0	0

for the stable species. The short-lived (free radical) species are initialized to zero since the photochemistry will achieve a mixing ratio for each of those species consistent with all other species in a short time. The initialization of the simulation is performed by running the model 2 or 3 days prior to the actual simulation period. The resulting concentration field is then used as the initial conditions of the simulation.

The horizontal boundary conditions used when solving Equation (5.1) are:

$$u_i c_i - K_{ii} \frac{\partial c_i}{\partial x_i} = F_{b,i}. \tag{5.5}$$

The horizontal flux, $F_{b,i}$ of species i, at the boundary is computed as

$$F_{b,i} = V_p c_{b,i}, \tag{5.6}$$

where V_p is the normal wind component at the boundary, $c_{b,i}$ is the boundary concentration during inflow conditions, or the concentration next to the boundary during

outflow conditions.

The vertical boundary conditions used when solving Equation (5.1) are:

$$K_{zz}\frac{\partial c_i}{\partial \sigma} = v_{d,i}c_i \qquad \text{at} \qquad \sigma = 0.995 \tag{5.7}$$

$$K_{zz}\frac{\partial c_i}{\partial \sigma} = 0 \qquad \text{at} \qquad \sigma = 0.05 \tag{5.8}$$

where $v_{d,i}$ is the deposition velocity of species i. Equation (5.8) implies a no-flux condition at the top of the modeling region. Consequently, the model is insensitive to upper boundary conditions for a species concentration.

The numerical solution of Equation (5.1) is not amenable to classical discretization techniques. Implicit discretization produces large sparse matrices which cannot be economically inverted. Explicit discretization requires prohibitively small time steps to advance the solution. Current chemical mechanism involve reaction times that differ by $O(10^{10})$ seconds. To overcome these problems, SARMAP uses splitting methods (Yanenko, 1971). Splitting methods produce a weak[1] solution to the problem but allows the use of numerical techniques that have been tailored to specific physical character of the operators used. The basic idea of the splitting process is the use of operators, $\mathcal{L}$, that describe horizontal transport:

$$\mathcal{L}_H c_i = \frac{\partial c_i}{\partial t} = -\nabla_H \cdot \mathbf{u}c_i + \nabla_H \cdot \mathbf{K}\nabla_H c_i, \tag{5.9}$$

vertical transport:

$$\mathcal{L}_z c_i = \frac{\partial c_i}{\partial t} = -\frac{\partial}{\partial z}(wc_i) + \frac{\partial}{\partial z}(K_{zz}\frac{\partial c_i}{\partial z}), \tag{5.10}$$

chemistry plus emissions:

$$\mathcal{L}_c c_i = \frac{\partial c_i}{\partial t} = R_i(\mathbf{c}, T) + S_i(\mathbf{x}, t), \tag{5.11}$$

[1] A weak approximation is a solution that satisfies only an integral from of the governing equation.

and cloud processes:

$$\mathcal{L}_{cl}c_i = \frac{\partial c_i}{\partial t}|_{clouds}. \tag{5.12}$$

The horizontal transport operator can be further decomposed into its x and y components. Then the temporal approximation of Equation (5.1) is obtained from the solution of the sequence:

$$\mathbf{c}^{t+2\Delta t} = \mathcal{L}_x^{\Delta t}\mathcal{L}_y^{\Delta t}\mathcal{L}_z^{\Delta t}\mathcal{L}_c^{2\Delta t}\mathcal{L}_{cl}^{2\Delta t}\mathcal{L}_z^{\Delta t}\mathcal{L}_y^{\Delta t}\mathcal{L}_x^{\Delta t}\mathbf{c}^t. \tag{5.13}$$

Figure 5.1 shows a flow diagram of the operator splitting solution used in SARMAP. Details of the numerical solution of the operators are given in the next section.

5.2 Numerical Implementation

5.2.1 Horizontal Advection

The solution of the advection operator when solving Equation (5.1) accounts for the transport of pollutants under a given wind field. The two-dimensional advection equation is

$$\frac{\partial C}{\partial t} + \frac{\partial (uC)}{\partial x} + \frac{\partial (vC)}{\partial y} = 0, \tag{5.14}$$

where C is the concentration, t is time, and u, v are the x, y components of the wind velocity field. Problems of numerical diffusion, peak concentration resolution, and spurious oscillations arise in the numerical solution of Equation (5.14) as both the amplitude and phase of different Fourier components of the solution tend to be altered by numerical schemes. Many authors have developed and/or tested a significant number of algorithms to be used in air pollution modeling. Two of the most attractive algorithms are the BOTT solver and the Galerkin finite element approach (GALK). This study implements and compares the use of BOTT and GALK in the SARMAP model. The main problem encountered in the solution of the advection operator is that of accuracy.

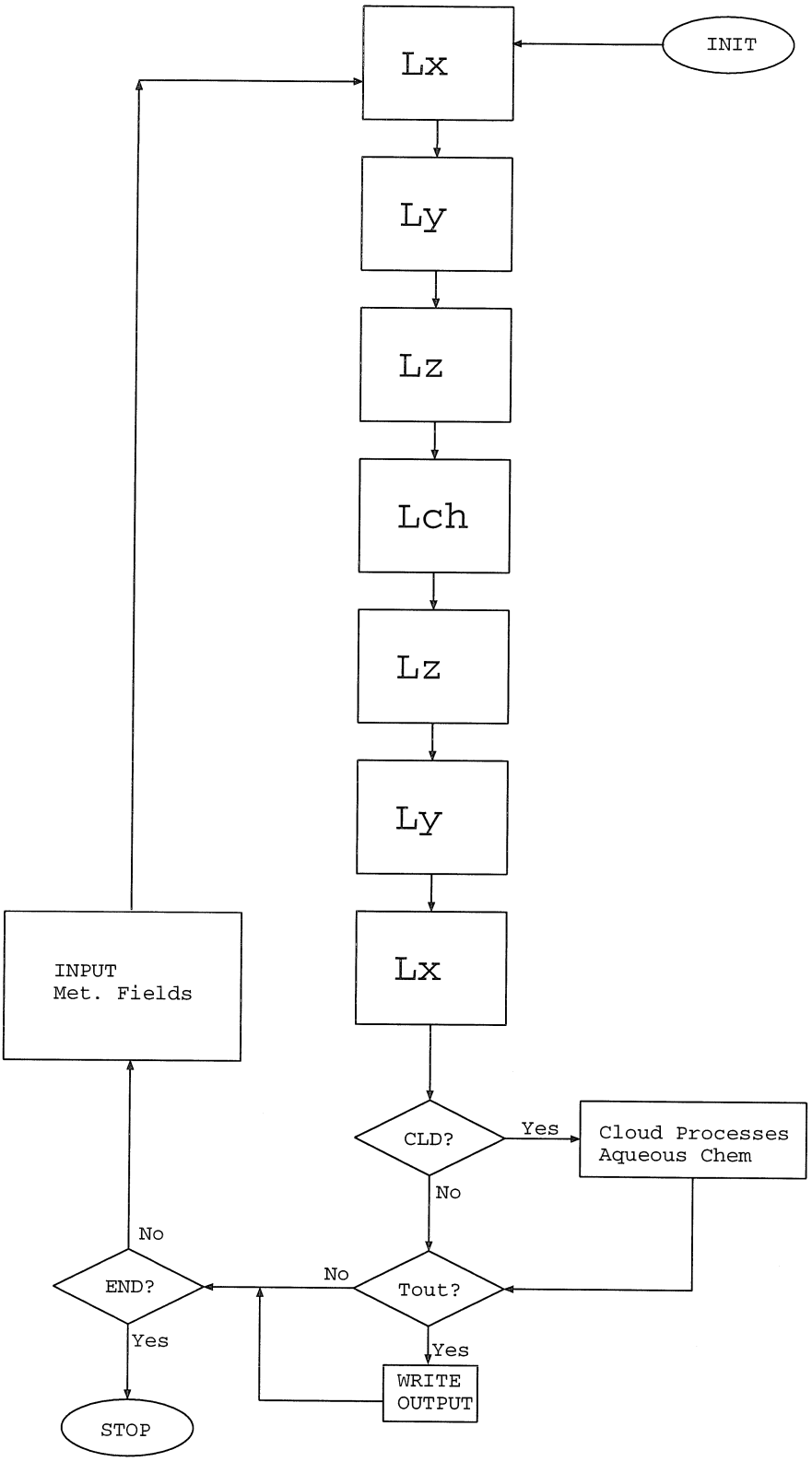


Figure 5.1: Schematic representation of the operator splitting method used by SARMAP.

The horizontal advection operator of the SARMAP model is solved using ADI techniques (Yanenko, 1971), that is, decomposing the two-dimensional problem described by Equation (5.14) by the successive solution of

$$\frac{\partial C}{\partial t} + \frac{\partial (uC)}{\partial x} = 0 \tag{5.15}$$

$$\frac{\partial C}{\partial t} + \frac{\partial (vC)}{\partial y} = 0. \tag{5.16}$$

BOTT (Bott, 1989a; Bott 1989b) develops further the polynomial fitting techniques proposed initially by Crowley (1968). The idea behind the solver is to approximate the concentration field by a fourth-order polynomial. The construction of the polynomial is guided by the curvature and magnitude of the concentration field. Fluxes are evaluated followed by their weighting and limitation to achieve positive definiteness and phase and amplitude error reduction. The Galerkin finite element approach (McRae *et al.*, 1982) uses a conventional Chapeau-function as the basis function of the solution space. It produces a tridiagonal system of equations that is solved with relatively small computational effort. GALK is not positive definite; i.e., it does not guarantee the preservation of positive values of its output. As a result, the GALK solution requires the use of filters which redistribute the non-physically negative mass predicted by the algorithm.

Figure 5.2 compares ozone concentrations observed in the San Joaquin Valley of California with those predicted by the model using BOTT and GALK solvers for the August 3-5, 1990. A detailed discussion of the episode is given subsequently. The simulation starts on August 2, 1990 at 0500 hrs. Ozone predictions using the two horizontal advection solver exhibit significant differences as early as August 3. It can be observed that BOTT presents better peak preservation properties and smaller artificial diffusion in regions of steep gradients where ozone levels are high. Such behavior confirms numerical experiments reported by Dabdub and Seinfeld (1994) that compare the solvers using simpler test cases. Moreover, it is observed that ozone levels in upwind locations (Sacramento, Stockton, Livermore, Crows Landing and

Gilroy) are over predicted by GALK. On the other hand, ozone levels in downwind locations (Arvin, Fresno) are under predicted by GALK. The Galerkin solver is not as efficient as the BOTT solver in advecting pollutants from upwind to downwind areas. The streamlines of the advective flow used in the simulation are presented in the next section. All the results shown in the rest of the chapter are obtained using BOTT as the solver for the horizontal advection operator.

5.2.2 Vertical Diffusion

Vertical mixing is modeled as an eddy diffusion process. The vertical component of the diffusion equation is discretized for level k as

$$c_k^{t+\Delta t} = c_k^t + \frac{\Delta t}{\Delta \sigma}[K'_{k+1/2}\{\theta(c_{k+1}^{t+\Delta t} - c_k^{t+\Delta t}) + (1-\theta)(c_{k+1}^t - c_k^t)\} \\ -K'_{k-1/2}\{\theta(c_k^{t+\Delta t} - c_{k-1}^{t+\Delta t}) + (1-\theta)(c_k^t - c_{k-1}^t)\}] \qquad (5.17)$$

where θ is the time weighting factor. SARMAP uses θ=1/2. Equation (5.17) is a semi-implicit (Crank-Nicholson) finite difference discretization. The discrete problem is reduced to the solution of a tridiagonal system which is solved using the Thomas algorithm. In a σ-coordinate system the eddy diffusivities are

$$K'_{k-1/2} = K^\star_{k-1/2}/\Delta\sigma_{k-1/2} \qquad (5.18)$$

where

$$K^\star = (g\rho/P^\star)^2 K_{zz} \qquad (5.19)$$

ρ is the density of air, and

$$\Delta\sigma_{k-1/2} = \Delta\sigma_k - \Delta\sigma_{k-1}. \qquad (5.20)$$

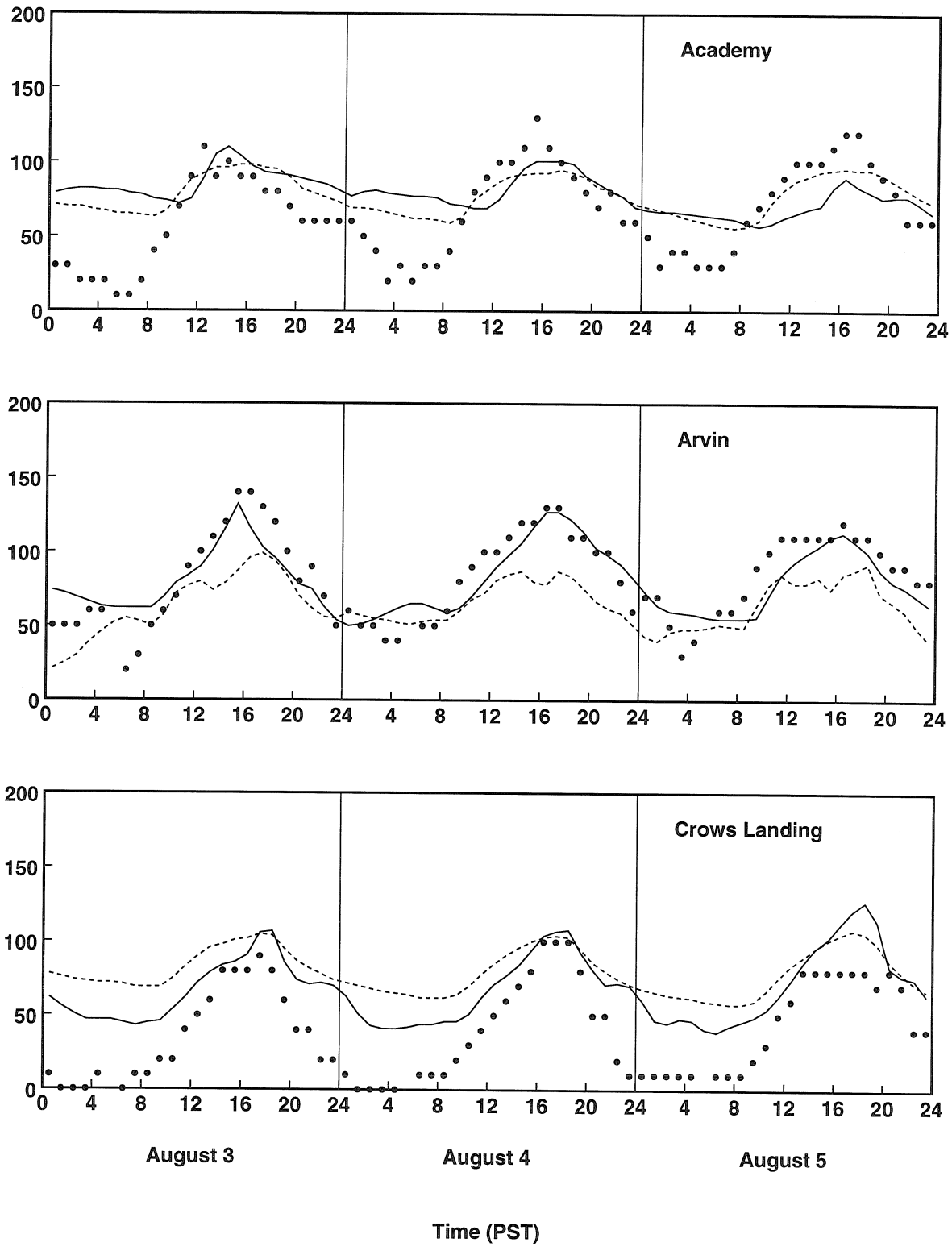


Figure 5.2: Time series plot of observed ozone concentrations in ppb (solid circles) and model predictions using BOTT solver (solid line) and Galerkin solver (dashed line).

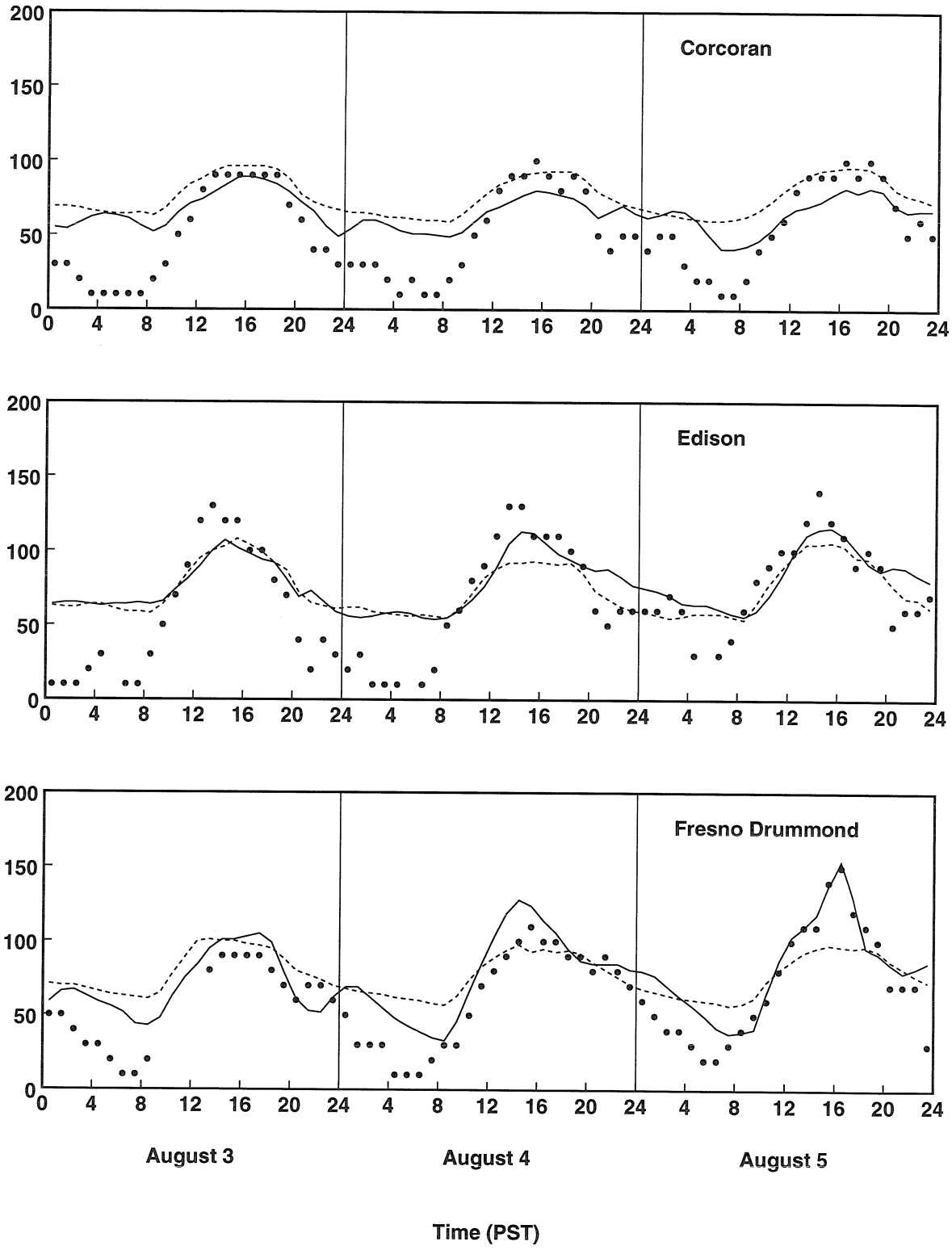


Figure 5.2: (Continuation)

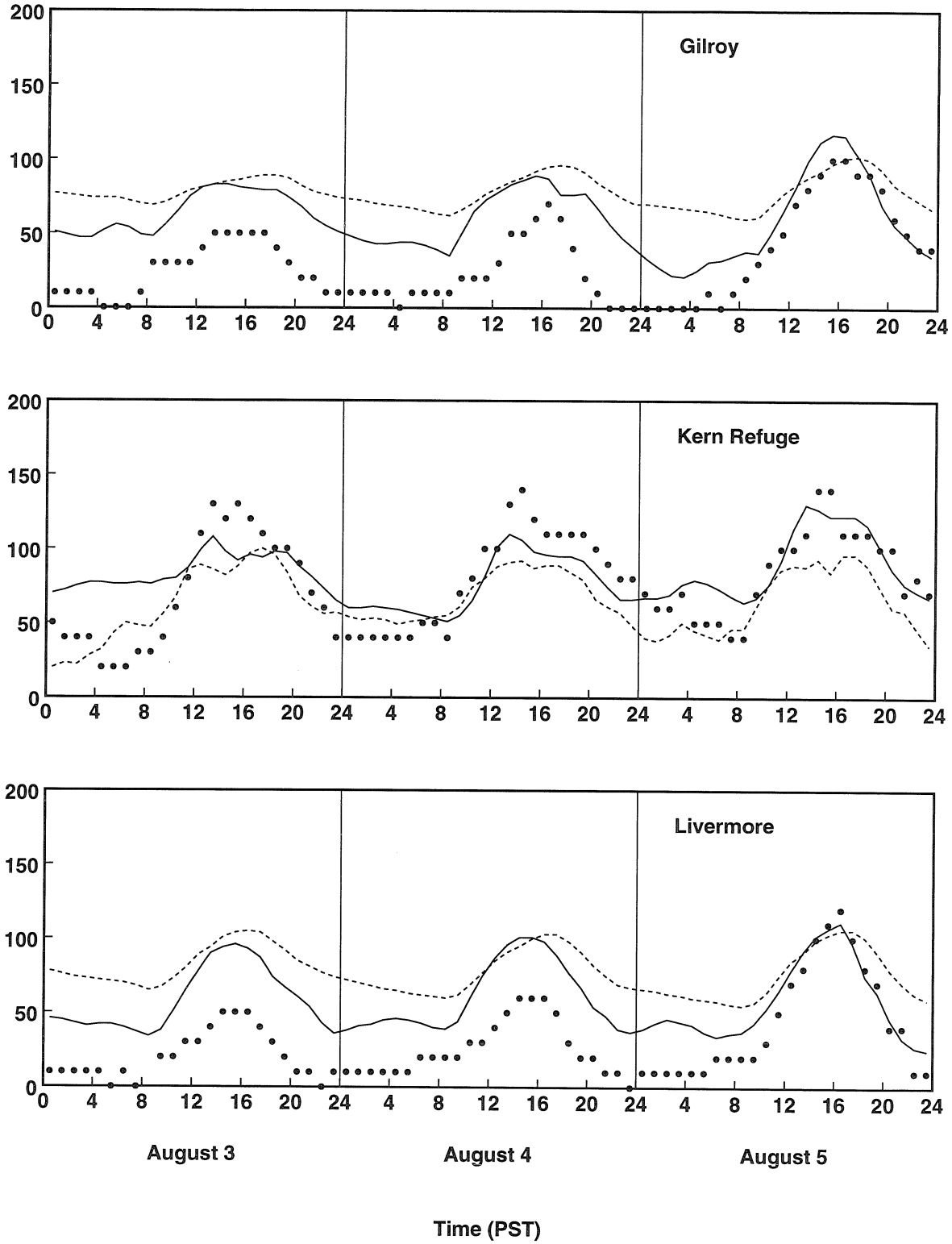


Figure 5.2: (Continuation)

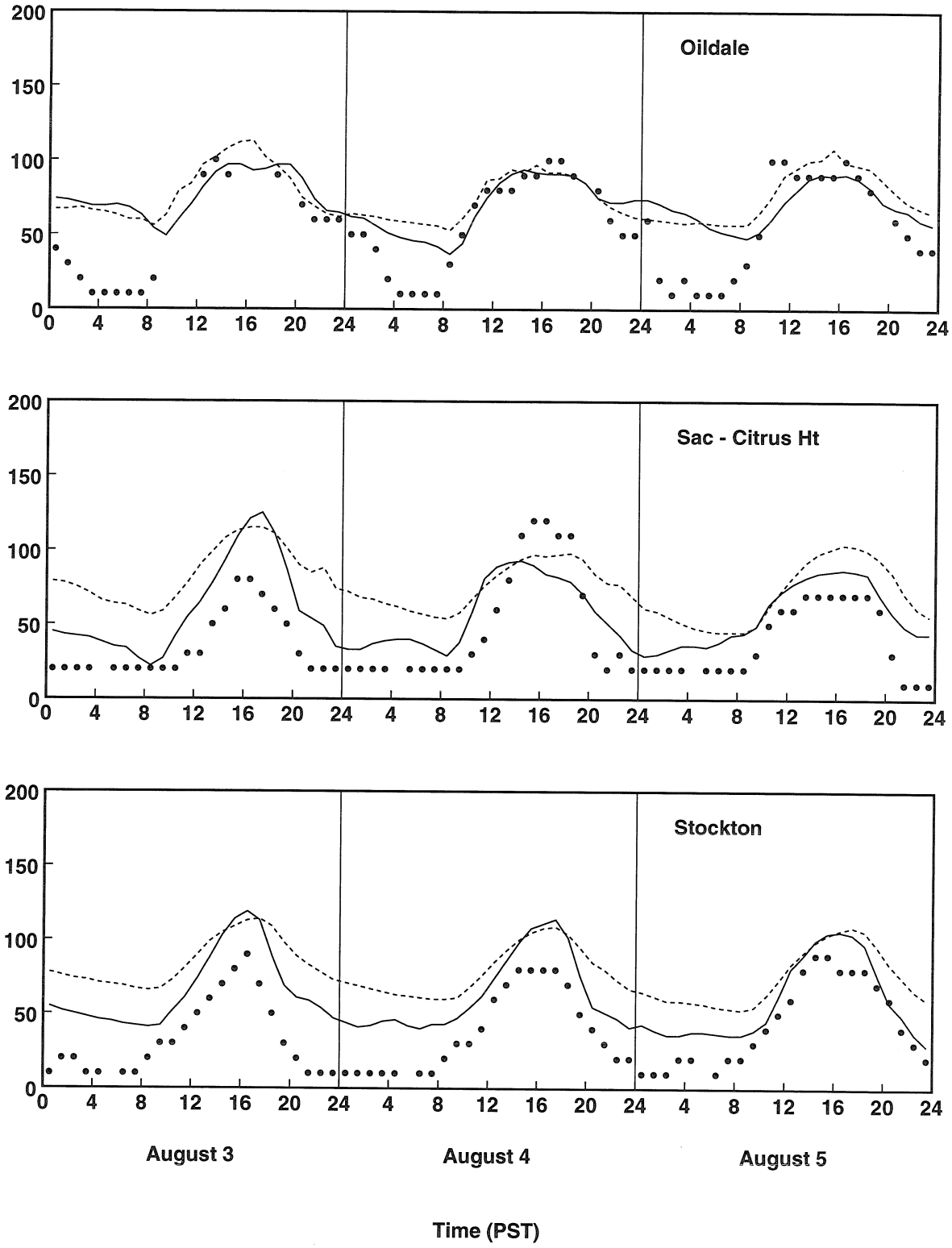


Figure 5.2: (Continuation)

5.2.3 Chemistry

The solution of the chemistry operator when solving Equation (5.1) accounts for the chemical transformation of pollutants within a given cell. The generic differential equation of chemical kinetics for each species i as included in SARMAP is expressed in the form

$$R_i(\mathbf{c}, T) = \frac{\partial c_i}{\partial t} = P_i - L_i = P_i - c_i/\tau_i \tag{5.21}$$

where P_i and L_i are the production and loss rates of species i. τ_i is the characteristic time scale for the first-order chemical loss. Equation (5.21) represents a system of nonlinear stiff ordinary differential equations. Simple stiff solvers would require extremely small steps to advance the solution of Equation (5.21) to maintain stability. General purpose solvers that implement variable order schemes, such as LSODE, require the computation of the Jacobian of the system. A set of algebraic equations must be solved at each time step, a relatively time-consuming operation. The main problem encountered in the solution of the chemistry operator is that of stability and speed. SARMAP uses the CBM4 chemical mechanism. The mechanism consists of 29 differential species in 83 reactions. Table 5.2 shows a list of the differential species defined in the chemical mechanism.

5.2.4 Photolysis Rates

Photolysis rates ($j-$values) for the photolytic reactions in SARMAP are calculated from a radiative transfer model (Joseph *et al.,* 1976) based on the delta-Eddington technique for a grid of 130 spectral intervals from 1 to 10 nm. The database of clear sky photolysis rates is the result of integrating the product of the absorption cross section and the photo-dissociation quantum yield over all wavelengths λ, and directions φ, θ.

$$j_i = \int_\lambda \sigma_i(\lambda)\phi_i(\lambda) \int_\varphi \int_\theta I(\lambda, \varphi, \theta) \sin\theta d\theta d\varphi d\lambda \tag{5.22}$$

where $\sigma_i(\lambda)$ is the molecular absorption cross section, $\phi_i(\lambda)$ the quantum yield for each photolysis reaction, and $I(\lambda, \varphi, \theta)$ the radiance at a particular location.

Table 5.2: Differential species defined in the CBM4 chemical mechanism.

Species code	Species name	Species code	Species name
SO2	Sulfur dioxide	SULF	Sulfuric acid
NO2	Nitrogen dioxide	NO	Nitric oxide
O3	Ozone	HNO3	Nitric acid
H2O2	Hydrogen peroxide	ALD	Acetaldehyde
HCHO	Formaldehyde	OPEN	High MW aromatic
PAR	Paraffins	ETH	Ethane
OLE	Olefins	ISO	Isoprene
NH3	Ammonia	N2O5	Nitrogen pentoxide
NO3	Nitrate radical	PAN	Peroxyacetylnitrate
TOL	Toluene	XYL	Xylene
CRES	Cresol	MGLY	Methylglyoxal
CO	Carbon monoxide	C2O3	Peroxyacyl radical
HONO	Nitrous acid	HNO4	Pernitric acid
CH4	Methane	HO	Hydroxyl radical
HO2	Hydroperoxyl radical		

The values of clear sky photolysis rate coefficients, j_{clear}, are calculated hourly between sunrise and sunset. The rates are stored at 0, 1, and 10 km in a database that is read by SARMAP. This database is used to perform linear interpolations to determine the $j-$values for each grid every 150 s during the simulation. The nature of the $j-$values is smooth enough so as not to be greatly affected by the interpolation procedure. For heights greater than 10 km photolysis rate coefficients for z=10 km are used.

To account for cloud coverage, the clear sky photolysis rates are corrected in the following manner:

$$j = j_{clear}(1 + a(F_{cld} - 1)) \tag{5.23}$$

where j is the corrected photolysis rate coefficient, a is the percentage of cloud coverage, and F_{cld} is the correction factor which depends on the location in the grid column. For solar zenith angles $\chi \leq 60°$

$$F_{cld} = 1 + \alpha_i(1 - t_r)\cos\chi \qquad \text{layer above cloud} \tag{5.24}$$

$$F_{cld} = 1.4 \cos \chi \qquad \text{layer in cloud} \tag{5.25}$$

$$F_{cld} = 1.6 t_r \cos \chi \qquad \text{layer below cloud} \tag{5.26}$$

where α_i is a tabulated reaction-dependent coefficient and t_r is the energy transmission coefficient for normally incident light.

$$t_r = (5 - e^{-\tau})/(r + 3\tau(1 - f)) \tag{5.27}$$

f is the scattering phase function asymmetry factor derived by Hansen and Travis (1974) and τ the cloud optical depth.

5.2.5 Dry Deposition

Dry deposition is the process by which pollutants are removed from the atmosphere at the earth surface. Dry deposition to surfaces (soil, water, or vegetation) is a significant sink for various trace species in the troposphere. Dry deposition fluxes are obtained in SARMAP as

$$\frac{\partial c_i}{\partial t} = v_{d,i} c_i / \Delta\sigma \tag{5.28}$$

where the deposition velocity, $v_{d,i}$, varies temporally and spatially. If the underlying land use contains different types of land, deposition velocities are a weighted function of land type for each grid cell,

$$v_{d,i} = \sum_s v_{d,s,i} f_s \tag{5.29}$$

for all s land types present, each with fraction f_s. The types of land categories used are: urban, agricultural, range, deciduous, forest, coniferous forest, forested swamp, water, swamp, and agricultural-range mixture.

Hourly dry deposition velocities are pre-processed from the land use database and MM5 generated data. SARMAP linearly interpolates the output of the processor every 150 s to compute the dry flux of a species. The preprocessor calculates the

deposition velocities from the following relation

$$v_{d,i} = (r_{a,i} + r_{b,i} + r_{c,i})^{-1} \tag{5.30}$$

where r_a is the aerodynamic resistance determined by turbulence in the surface layer, r_b is the resistance that controls transport through the laminar sub-layer in contact with the surface, and r_c is the surface resistance determined by season, land type, and insolation.

The aerodynamic resistance is parameterized in terms of the friction velocity and surface roughness over each land (Sheih *et al.*, 1979).

$$r_a = \frac{\ln(z/z_0 - \Psi_c)}{k u_\star} \tag{5.31}$$

where the friction velocity is defined as

$$u_\star = ku(\ln(z/z_0) - \Psi_m)^{-1}, \tag{5.32}$$

k is the von Karman constant, u the mean wind speed at height z, and z_0 the surface roughness scale length. Ψ_m and Ψ_c are stability correction functions that are a function of the Obukhov scale length, L. For stably stratified flow ($0 < z/L < 1$) (Dyer,1974) the stability functions are computed as

$$\Psi_m = \Psi_c = -5z/L. \tag{5.33}$$

For unstable conditions Wesely and Hicks (1977) suggest

$$\Psi_m = \exp\{0.032 + 0.448\ln(-z/L) - 0.132[\ln(-z/L)]^2\} \tag{5.34}$$

$$\Psi_c = \exp\{0.598 + 0.390\ln(-z/L) - 0.090[\ln(-z/L)]^2\}. \tag{5.35}$$

The resistance to transport across the laminar sub-layer is computed using the thermal

Table 5.3: Values of deposition velocities at noon on August 3, 1990 in Fresno.

Species code	Deposition Velocity (cm/s)
SO2	0.28
SULF	0.58
NO2	0.20
NO	0.16
O3	0.33
HNO3	1.44
H2O2	0.40
ALD	0.13
HCHO	0.22
PAR	0.20
OLE	0.17
NH3	0.91
PAN	0.29

(κ) and molecular (D_c) diffusivities of the species (Wesely and Hicks, 1977).

$$r_b = \frac{2(\kappa/D_c)^{2/3}}{\kappa u_\star}. \tag{5.36}$$

The surface resistance values are obtained from the values reported by Walcek *et al.* (1986). Surface resistances are tabulated by season and photosynthetic activity. Table 5.3 shows values of $v_{d,i}$ for all the species being deposited at noon in August 3, 1990 in Fresno.

5.2.6 Cloud Processes

The cloud processes subroutine has an area resolution of 80 km over a specified geographical region. The subroutine is executed whenever clouds or rain are predicted above a given grid cell during the simulation. Cloud effects account for vertical redistribution of pollutants, aqueous chemistry, and wet deposition.

The mixing rations are determined at each cloud level using the formulation em-

ployed by Chang *et al.* (1987)

$$\mu_{cld}(z) = f_{ent}[(1 - f_{side})\mu_{dw} + f_{side}\mu(z)] + (1 - f_{ent})\mu_{up} \tag{5.37}$$

where μ_{up} and μ_{dw} and upward and downward mixing ratios, f_{ent} and f_{side} are the fraction of air entrained into the parcel from above top and sides of the cloud respectively. Currently SARMAP uses f_{side}=0.

This submodel then averages the pollutant concentrations, liquid water content ($\overline{L}$), temperature ($\overline{T}$), and pressure ($\overline{P}$) over the entire cloud volume. The concentration of dissolved pollutants in the cloud water are computed as

$$\frac{\partial \mu_{cld}}{\partial t} = [\frac{\partial c_l}{\partial t} - c_l P_r/(\tau_{cld} f \Delta z_{cld} L_{con})]\overline{LT}R/\overline{P} \tag{5.38}$$

where R is the universal gas constant, f the fraction of cloud coverage, Δz_{cld} the cloud depth, τ_{cld} the cloud lifetime, P_r the grid average precipitation rate, L_{con} the total condensed water content, and $\frac{\partial c_l}{\partial t}$ the rate of change of species due to aqueous reactions. The aqueous phase chemical mechanism used is that described by Walcek and Taylor (1986).

Finally wet deposition is computed by integrating

$$\text{wet deposition} = \int_0^{\tau_{cld}} c_l P_r dt \quad \text{mol m}^{-2}. \tag{5.39}$$

5.3 Aspects of the Parallelization of the Model

A profile of the code, as shown in Table 5.4, indicates that the single most computationally intensive task in the model is the solution of the chemistry operator. Numerically, this is equivalent to the the integration of a system of stiff ODE's as described on previous sections. Furthermore, such operation is dependent only on local data. That is, the predicted pollutant concentration after a chemistry step is dependent on the concentration of other species that are located in the same grid cell. Both observations suggests that an initial attempt to parallelize the code should focus

Table 5.4: Summary of SARMAP profile for a base case episode.

Process	% of CPU time
Gas-phase chemistry	71.34
Vertical Advection & Diffusion	12.83
Horizontal Advection & Diffusion	8.74
Other	7.09

on distributing the integration of all the grid cells equally among all the processors.

The computing paradigm used to implement the parallelization strategy described below is host/slave. The role of the host is to perform the I/O, to solve the horizontal and vertical transport operators, to manage the data distribution used by the slaves. The role of the slaves is to receive the input data from the host, and to perform the computationally intensive work of solving the system of nonlinear ODEs that correspond to the chemistry operator. The communication channels required to following this approach are only between the host and each slave. There is no inter-node communication needed to parallize the chemistry operator. A parallel version of the code was developed to run on a network of 7 IBM RISC 6000/390 workstations. The workstations are networked using CDDI to form a distributed memory MIMD (multiple instruction/multiple data) computing platform. The message passing was performed using the P4 libraries and FORTRAN. Figure 5.3 shows the time to perform a 24-hour standard simulation of the San Joaquin Valley of California using SARMAP. The theoretical curve plotted in Figure 5.3 is obtained using Amdahl's law, and corresponds to best possible time that can be obtained implementing the chemistry in parallel only.

$$S = \frac{1}{(1-p) + p/N} \tag{5.40}$$

where S is the ideal speed-up, N the number of processors, and p the fraction of the code that is implemented in parallel. The theoretical time correspond to the best possible time since it assumes that all the communication costs between the host and nodes are instantaneous. In practice, however, communication costs are quite significant. The difference time shown in Figure 5.3 between the sequential code

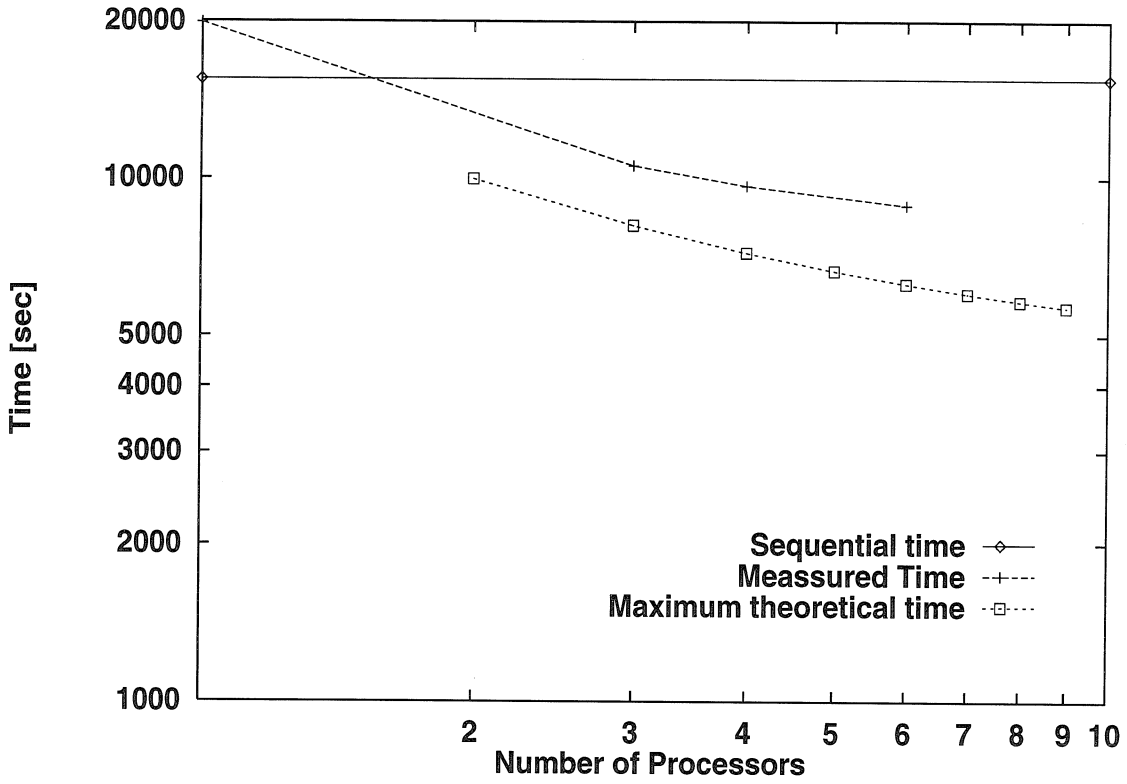


Figure 5.3: Execution time as a function of number of nodes when parallelizing the chemistry loop of SARMAP. Theoretical time presented is calculated from Amdahl's law. Time reported corresponds to a 24-hour standard simulation of the San Joaquin Valley of California.

and the parallel version using 1 processor corresponds to the time cost of all model communication.

Figure 5.3 shows that this regional atmospheric model requires greater internodal communication than the urban scale photochemical model. The number of grid cells in a typical regional model is about 5 times greater than the ones used in urban scale models. Furthermore, the computations carried out in SARMAP are performed using double precision arithmetic. As a result, there is a significant increase in the data flow of the computations.

Based on the previous discussion, the use of a faster network would substantially increase the performance of the parallel implementation. However, higher performance could also be achieved by implementing in parallel the solution of the vertical

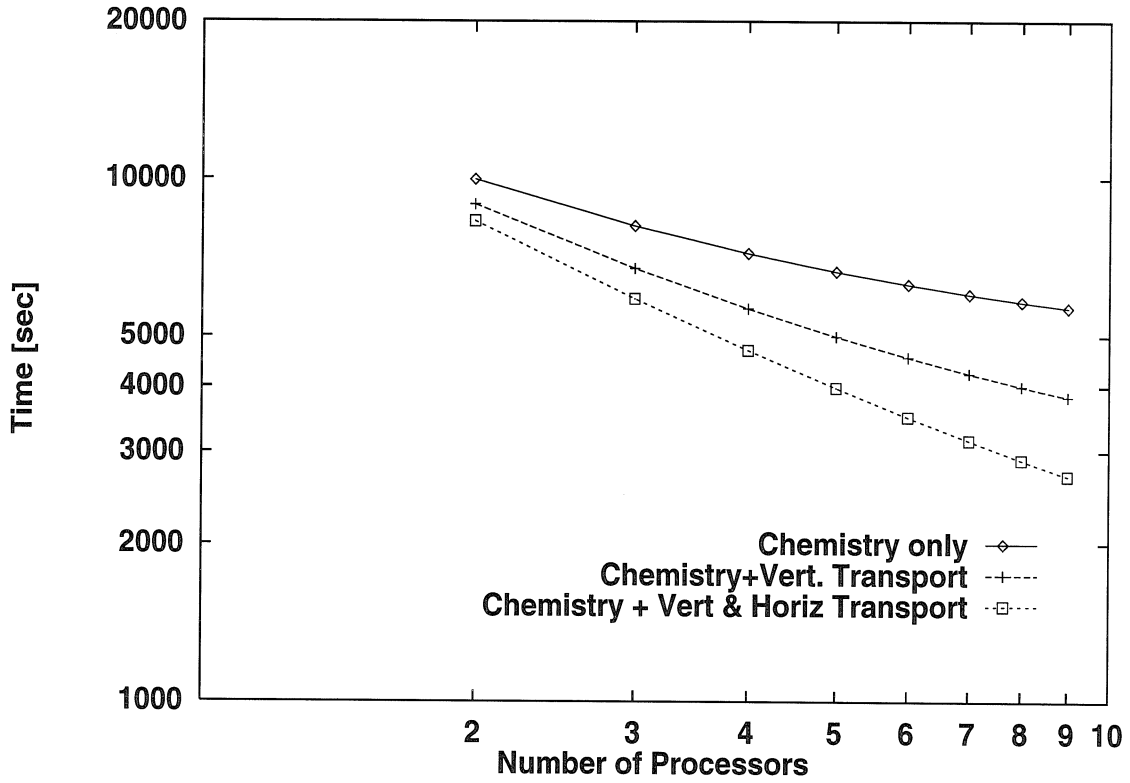


Figure 5.4: Execution time as a function of number of nodes when parallelizing various operators of the SARMAP model. Theoretical time presented is calculated from Amdahl's law. Time reported corresponds to a 24-hour standard simulation of the San Joaquin Valley of California.

and horizontal transport operators. Figure 5.4 shows a comparison of the theoretical times obtained by implementing in parallel the horizontal and vertical operators of the model, which is equivalent to increasing the value of p in Equation 5.40. Notice that the impact of perturbing p on the performance of code is proportional to the value of p before the perturbation.

In brief, the implementation of the chemistry operator of atmospheric dynamical models is a relatively simple task. It does not involve any communication among the slave nodes. The parallel implementation of the chemistry, as described above, is independent of the numerical scheme and the photochemical mechanism used.

5.4 Description of San Joaquin Valley Episode

In this section the application of the SARMAP model is described for a 3-day period during August 3-5, 1990. The modeling region is the San Joaquin Valley of California. The latitude and longitude of the the southwest and northeast corner of the modeling domain are (34.54472,-122.89846) and (39.08257,-118.21475) respectively. Figure 5.5 shows the location of a few air quality monitoring sites of the modeling region. The domain is divided into a 12.5×12.5 regular grid in the horizontal direction. As noted above, the model uses a σ-coordinate system in the vertical direction.

SARMAP requires hourly pollutant emission data for each grid cell. Each entry in the emissions inventory database contains the strength of the emission (in g/sec), the emitted species, and the time variation for each source for each grid. The emission inventory input file process a raw emissions database using a lumping procedure that is determined by the chemical mechanism used in the model. The species emitted in the simulation are SO2, SULF, NO2, NO, ALD, HCHO, PAR, ETH, OLE, ISO, TOL, XYL, CO, and HONO. The emission inventory used to simulate the August 1990 episode was created using GEMAP (Geocoded Emissions Modeling and Projection). The input data for GEMAP were obtained from the Air Resources Board emission inventory section. Figure 5.6 shows the time series plot for the CO, NO, and TOL emissions in Fresno. It can be easily observed that the emissions inventory is day-

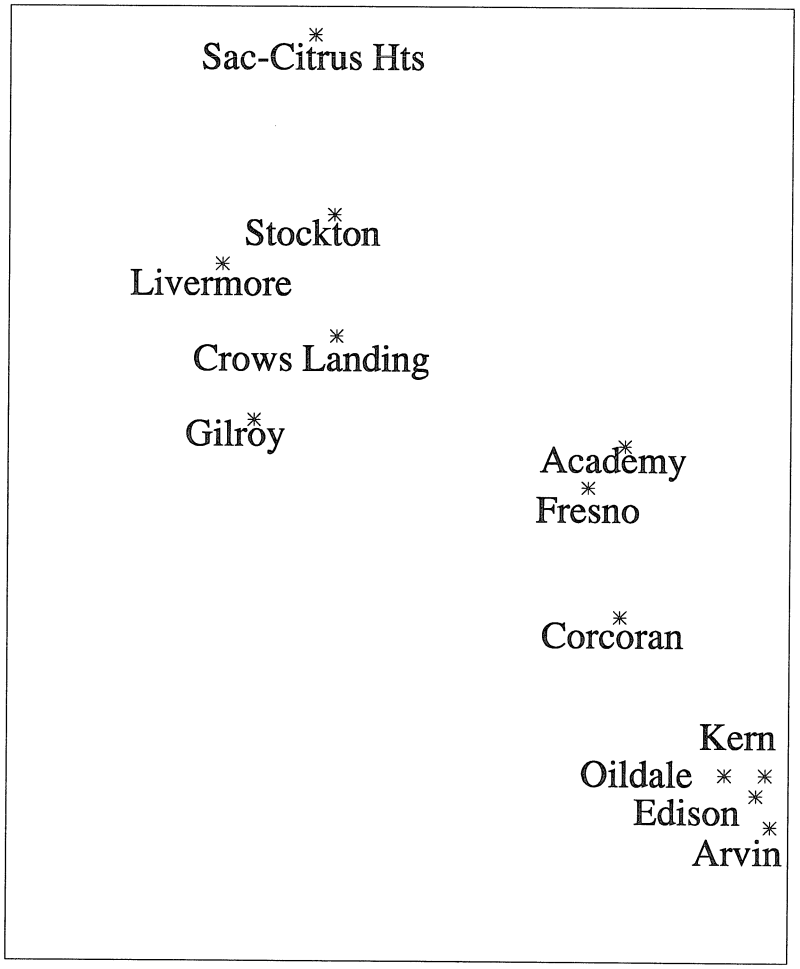


Figure 5.5: Location of some air quality monitoring sites of the modeling region.

Figure 5.5: (Continued)

specific. Emissions for the species shown are greater on August 3, which is a Friday. August 3 presents peaks at 0700 and 1600 hrs, the time of greatest mobile sources activity. Figures 5.7, 5.8 and 5.9 show the ISO, NO and TOL emissions contour plots respectively at 1200 hrs. ISO emissions are higher in areas of dense vegetation. The Sierra Nevada and the Santa Lucia range form the two main diagonals of high ISO emissions shown in Figure 5.7. Emissions of NO and TOL are associated with motor vehicle activity. Figures 5.8 and 5.9 show that urban centers (San Francisco, San Jose, Sacramento, and Fresno) present the highest NO and TOL emissions. Figure 5.10 shows the total rate of emission for each species in the entire modeling region. The first hour modeled is at 0500 hrs on Thursday August 2, 1990. The rate of emissions of some species (SO2, SULF, NO2, and NO among others) show the day-specificity of the inventory. Furthermore, emission rates of species like CO and NO2 illustrate the hourly emission variations inherent to traffic patterns. On the other hand, the emission rate of ISO increases significantly during last two days of the simulation. Such increase is the effect of a higher temperature in the region modeled.

Whereas the time discretization used in SARMAP is comprised of a basic time step of 300 s for each operator, each operator has control of its internal time discretization. The time step used by the advection, diffusion, and dry deposition calculations is typically half of the inter-operator time step. On the other hand, the time step used in the chemistry operator is significantly smaller and highly dependent on the degree of stiffness of the ODEs that describe the chemistry. Periods of sunset and sunrise present the highest degree of stiffness, when rapid changes occur in the concentrations of photochemically driven species.

The flow field $\mathbf{u}(\mathbf{x}, t)$ is provided as input to Equation (5.1) from mesoscale meteorological models such as MM5 (Grell, 1993). MM5 provides the three-dimensional advective field from fully predictive, nudged, or objectively interpolated computations (Seaman *et al., 1995*). In addition to the advective field, MM5 provides three-dimensional temperature fields. Figures 5.11, 5.12 and 5.13 show the streamlines of the horizontal advective flow in the lowest grid cells used to simulated the August, 1990 episode in the San Joaquin Valley of California. It is observed that during all

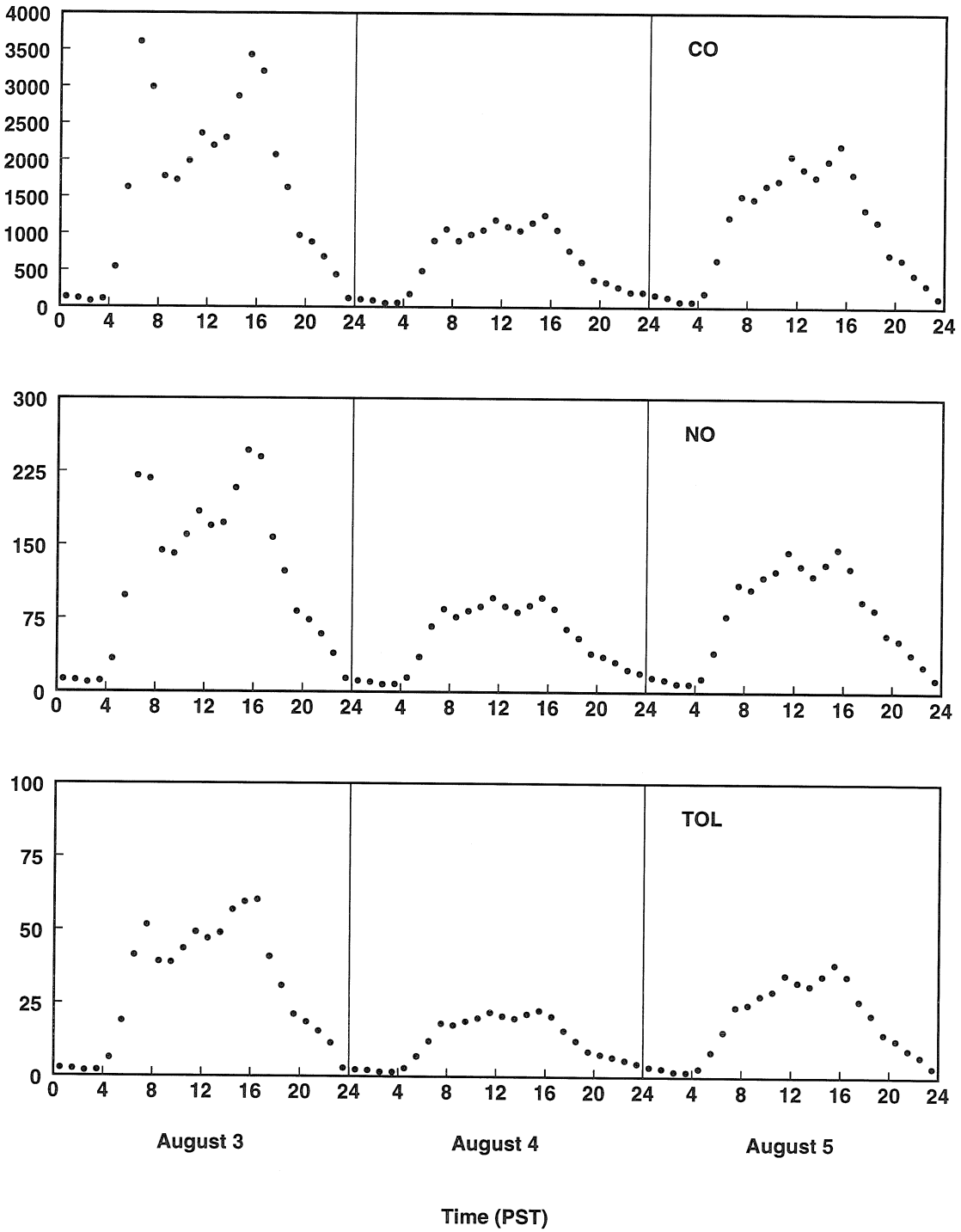


Figure 5.6: Time series plot of the CO, NO, and TOL emissions in g/s in Fresno used in the simulation of the August 1990 episode.

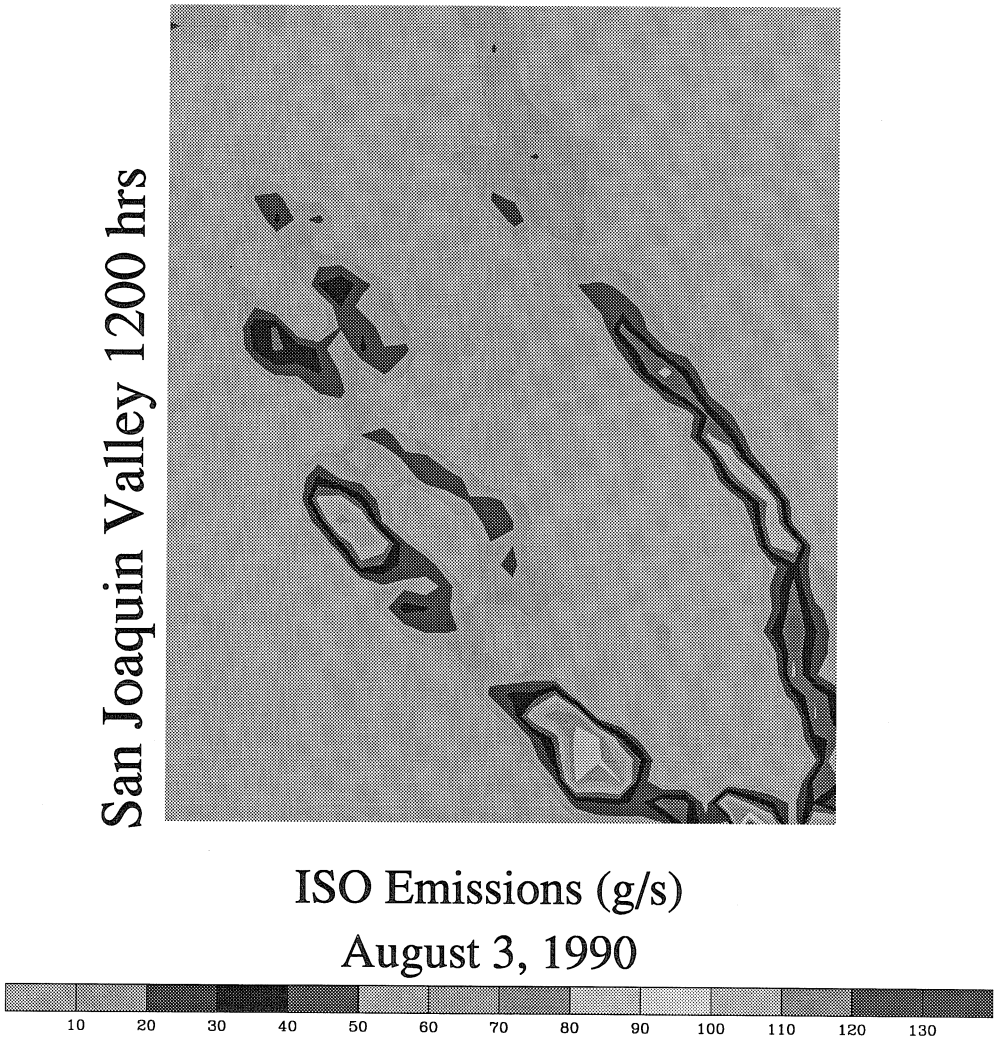


Figure 5.7: ISO emissions at 1200 hrs on August 3, 1990.

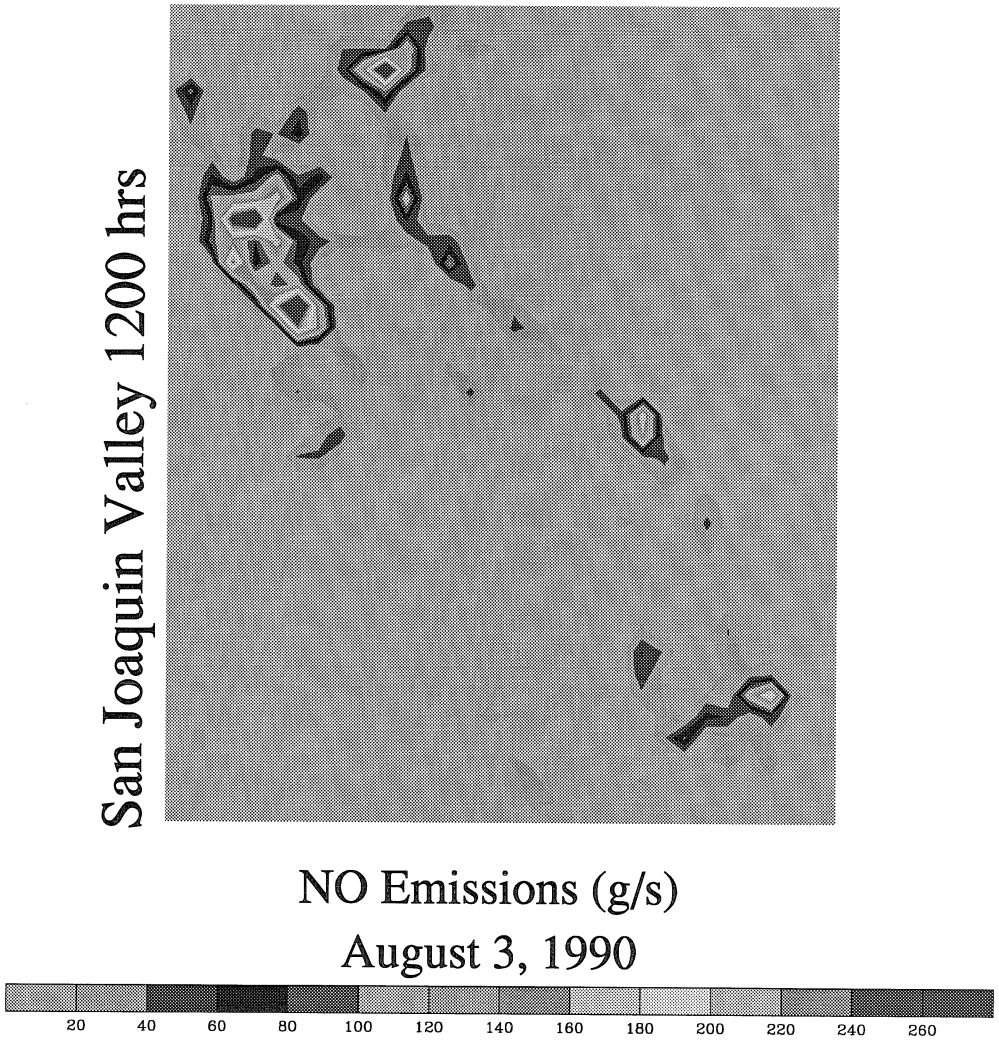


Figure 5.8: NO emissions at 1200 hrs on August 3, 1990.

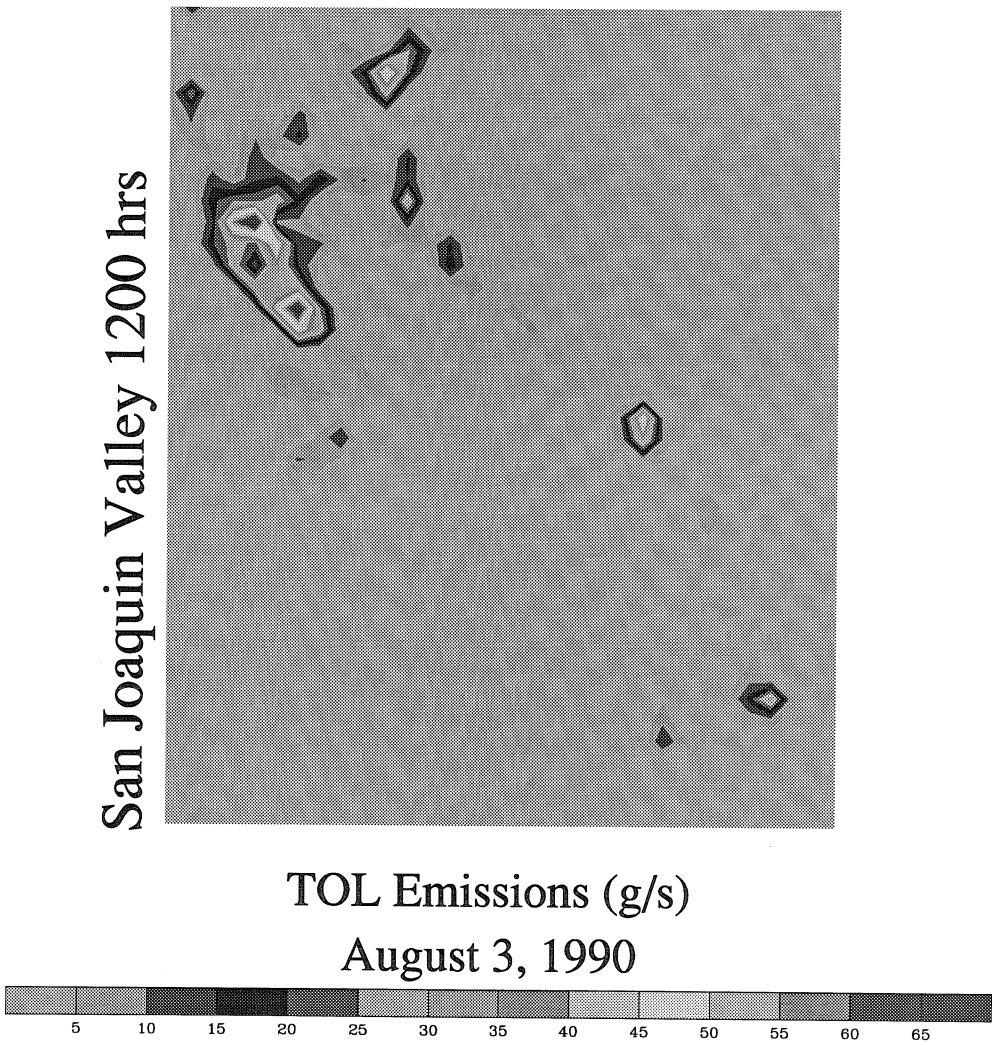


Figure 5.9: TOL emissions at 1200 hrs on August 3, 1990.

times of the simulation the northern part of the eastern modeling region present influx conditions. As a result, one would expect to observe a significant influence of the boundary conditions on nearby sites.

MM5 specifies the projection type used to map the advective and temperature fields to SARMAP. The projections are necessary to transform the constant grid ($\Delta s = \Delta x = \Delta y$) used by MM5 to the actual distance on the earth's surface ($\Delta s'$) used by SARMAP. The transformation factor is $m = \Delta s/\Delta s'$. Currently available projection types are: polar stereographic, Lambert conformal, and the Mercator projection.

The Lambert conformal conic projection is used for middle latitudes studies.

$$X = A[\tan(\Psi/2)]^s \cos[s(\lambda - \lambda_0) - \pi/2] \tag{5.41}$$

$$Y = A[(\tan(\Psi_0/2))^s(\tan(\Psi/2))^s \sin(s(\lambda - \lambda_0) - \pi/2)] \tag{5.42}$$

where Ψ is the colatitude, $\Psi = \pi/2 - \varphi$. The origin of the projection is (latitude,longitude) $= (\varphi_0, \lambda_0) = (X, Y)$. The coefficients A and s are defined as

$$A = \frac{R \sin \Psi_1}{s(\tan(\Psi_1/2))^s} \tag{5.43}$$

$$s = \frac{\ln(\cos \varphi_1 / \cos \varphi_2)}{\ln(\tan(\Psi_1/2) / \tan(\Psi_2/2))}. \tag{5.44}$$

R is the radius of the earth, Ψ_1 and Ψ_2 are computed for mid-latitudes from $\varphi_1 = 30°$N and $\varphi_2 = 60°$N. The transformation factor is given by

$$m = \sin \Psi_1 / \sin \Psi [\tan(\Psi/2) / \tan(\Psi_1/2)]^s. \tag{5.45}$$

The polar stereographic projection is preferred for high latitudes. The values of (φ, λ) are obtained as in the Lambert projection. However, the transformation factor is given by

$$m = (1 + \sin \varphi_1)/(1 + \sin \varphi). \tag{5.46}$$

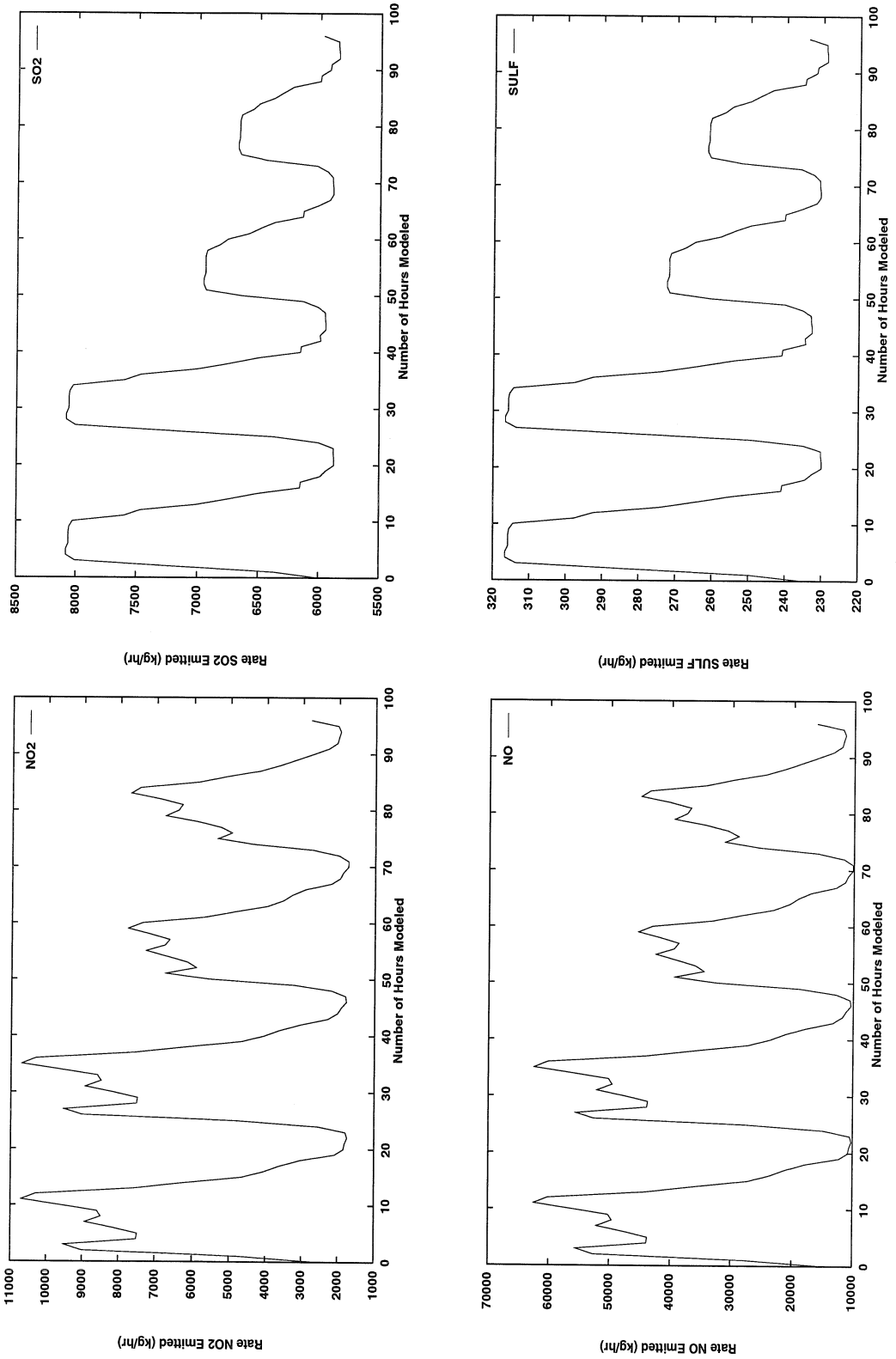


Figure 5.10: Total emission rates of the entire modeling domain during the August 3-5, 1990 episode of the San Joaquin Valley of California.

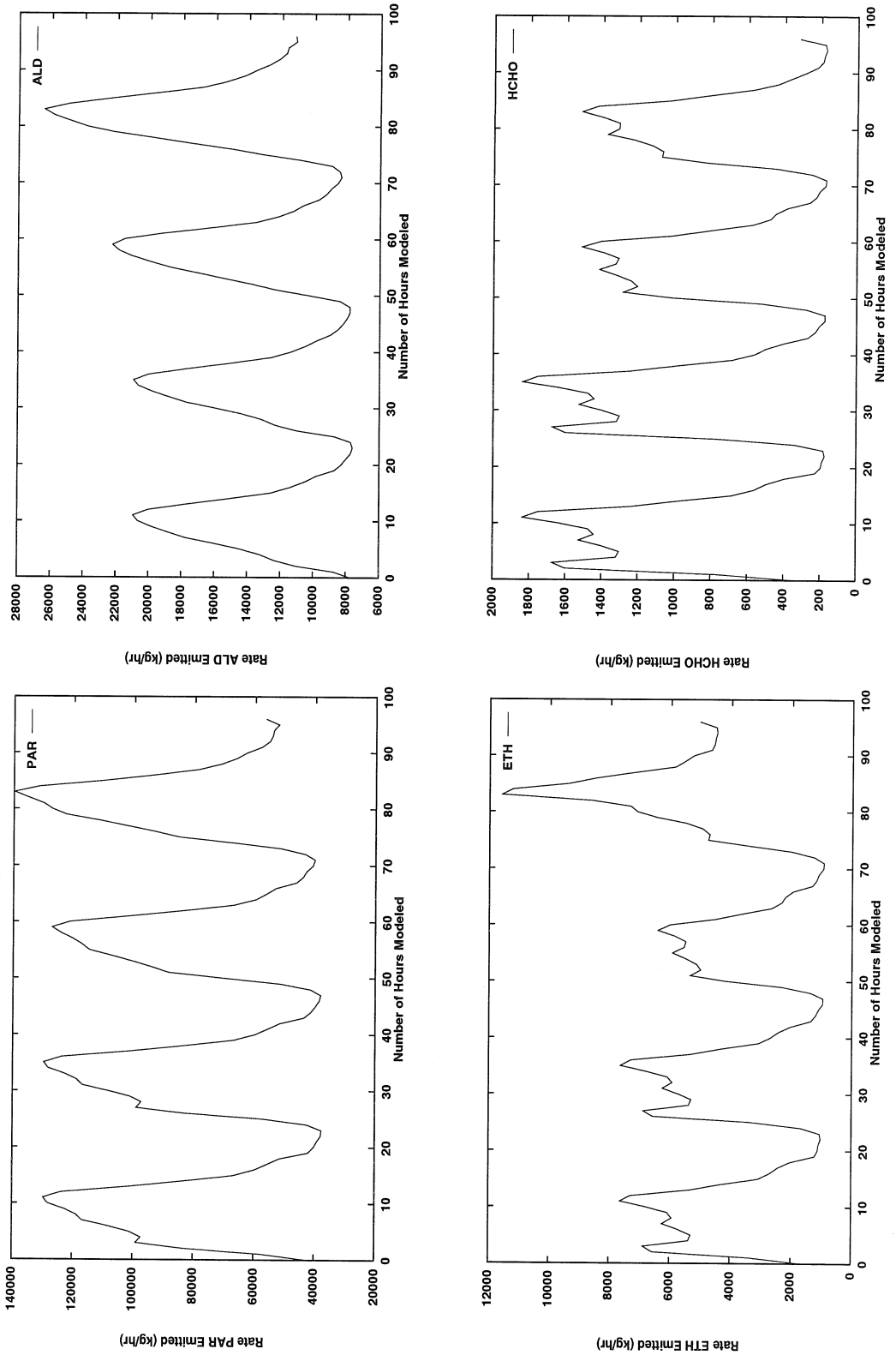


Figure 5.10: (Continuation)

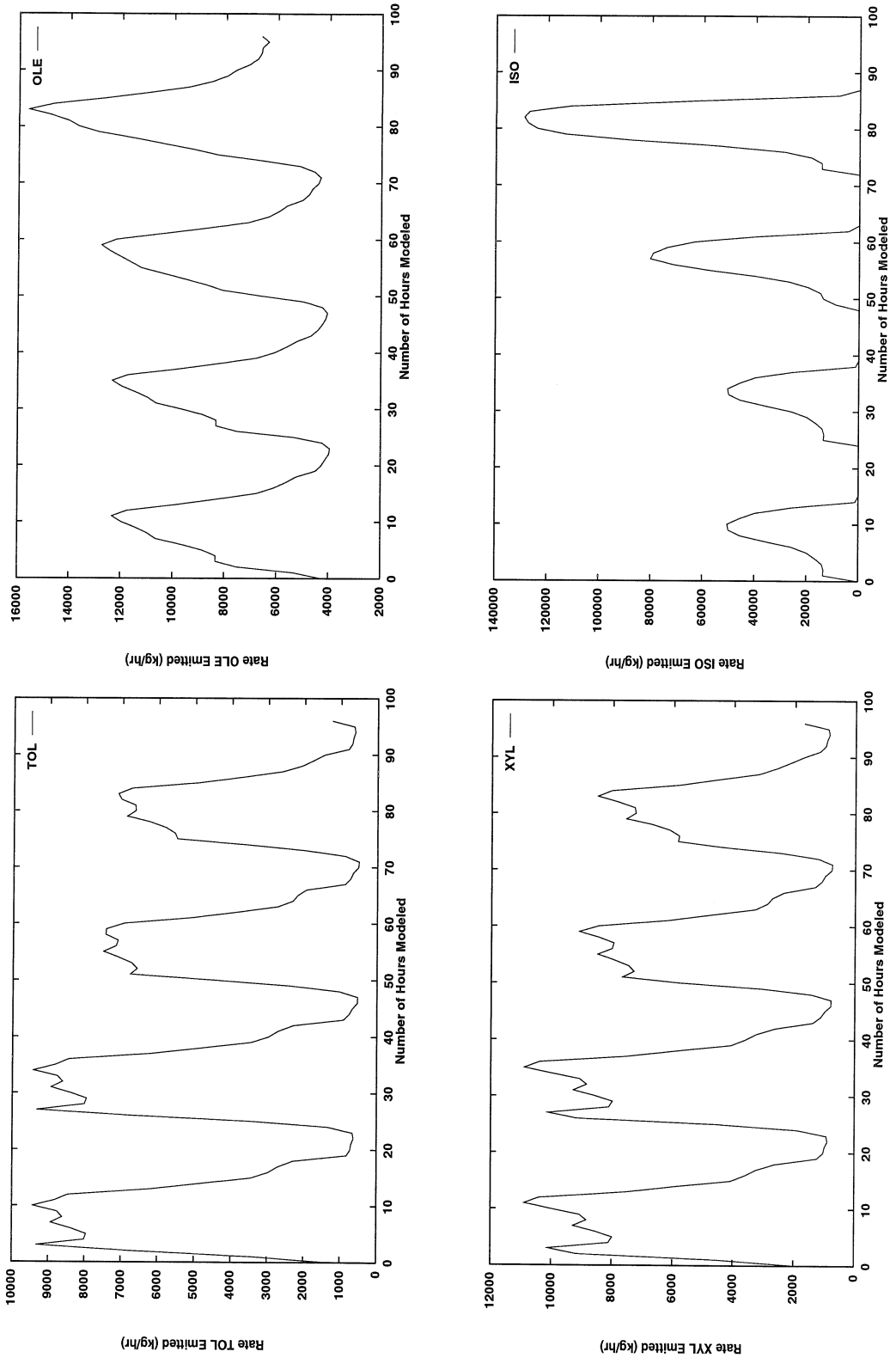


Figure 5.10: (Continuation)

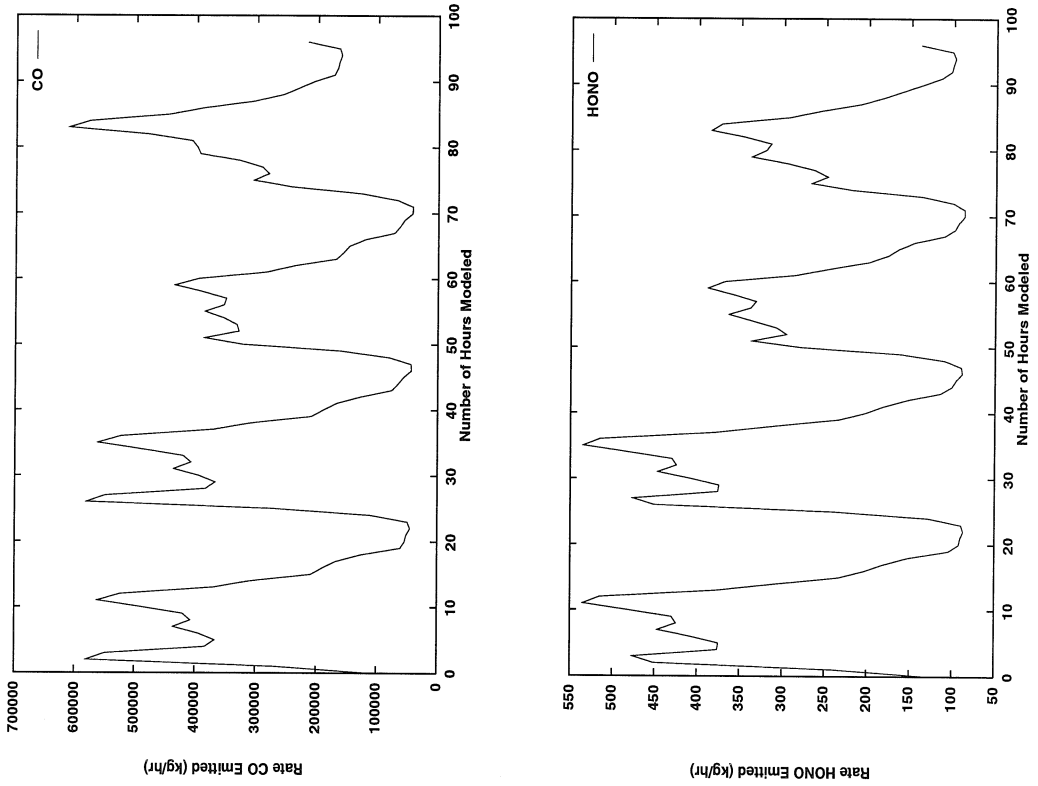


Figure 5.10: (Continuation)

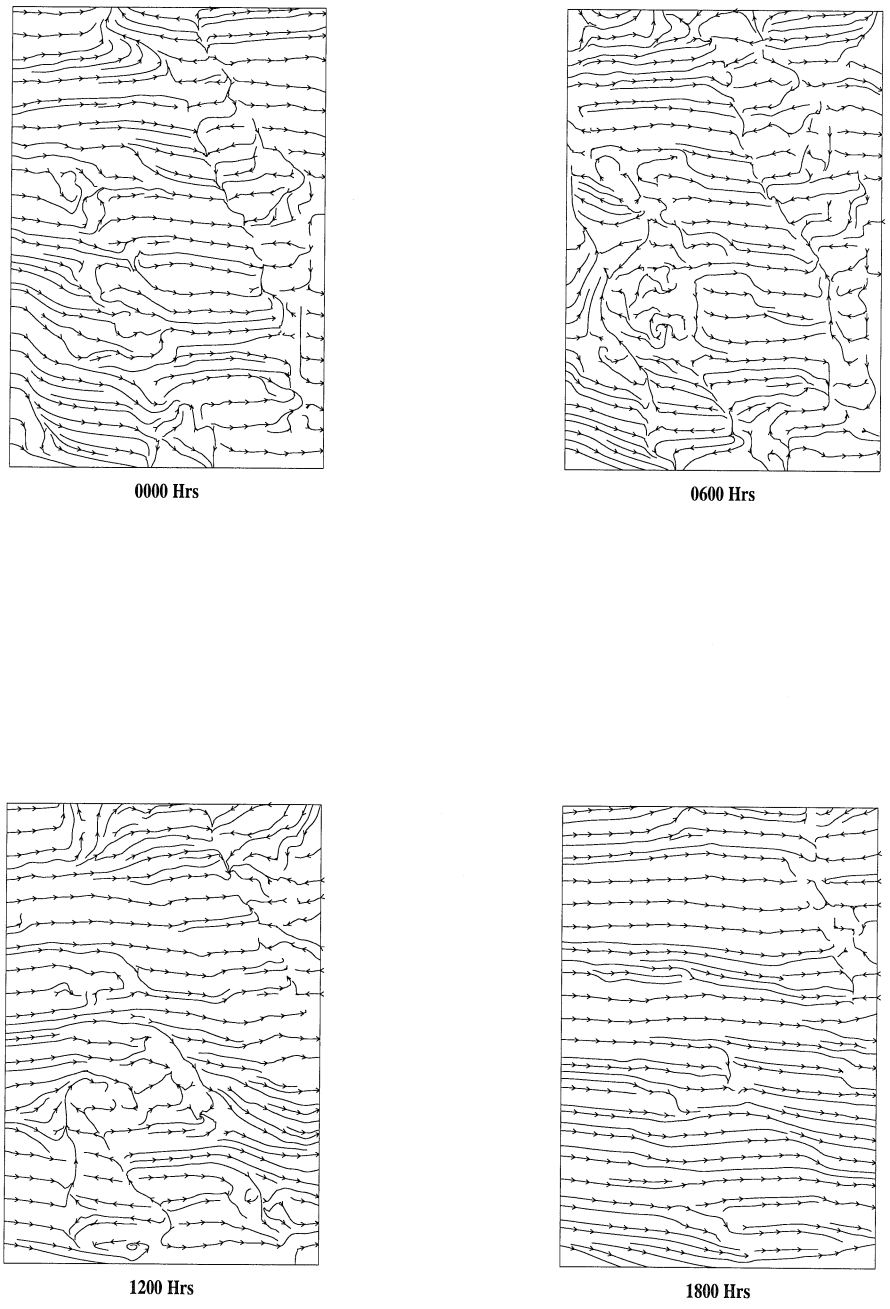


Figure 5.11: Streamlines of the horizontal advective flow field used in the bottom vertical layer for August 3, 1990

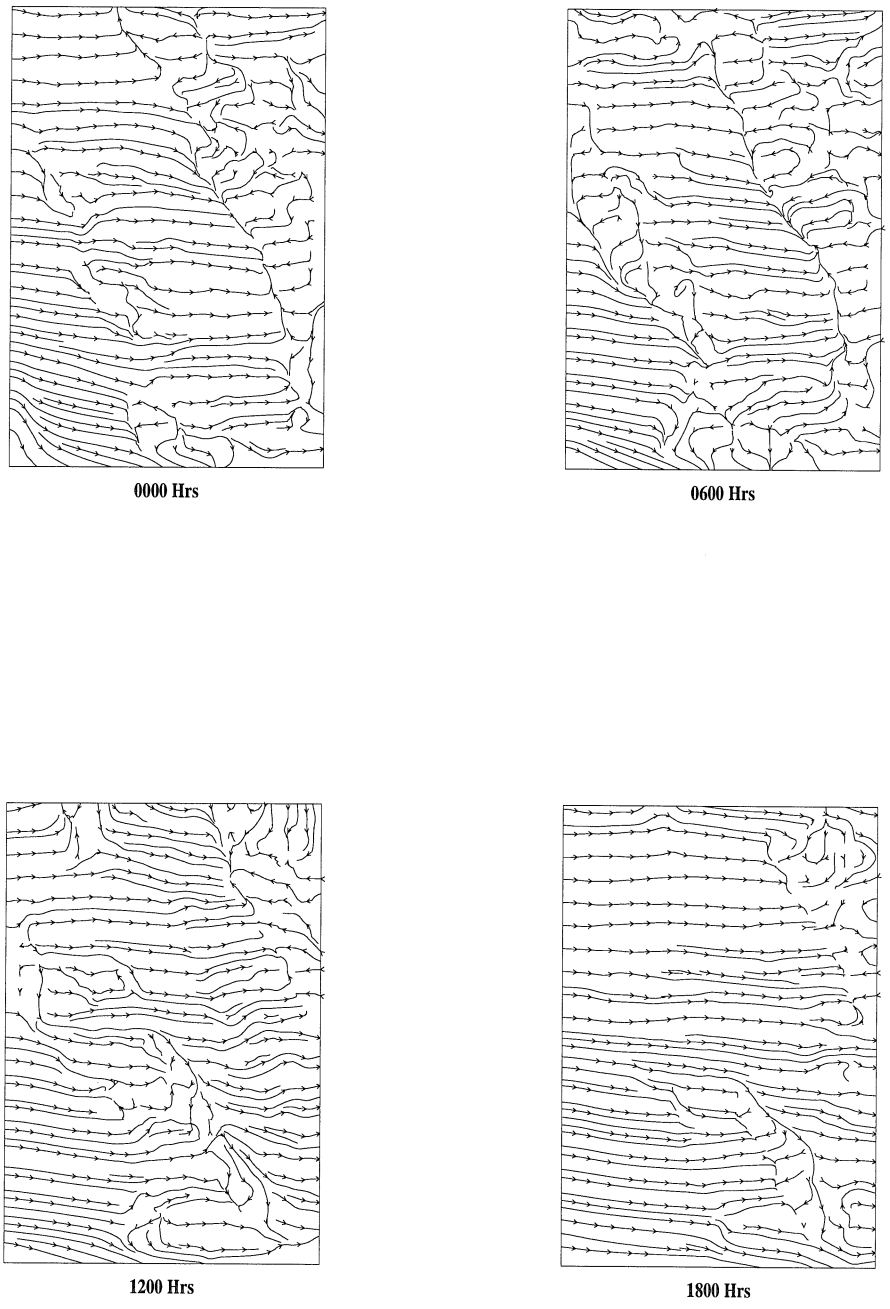


Figure 5.12: Streamlines of the horizontal advective flow field used in the bottom vertical layer for August 4, 1990

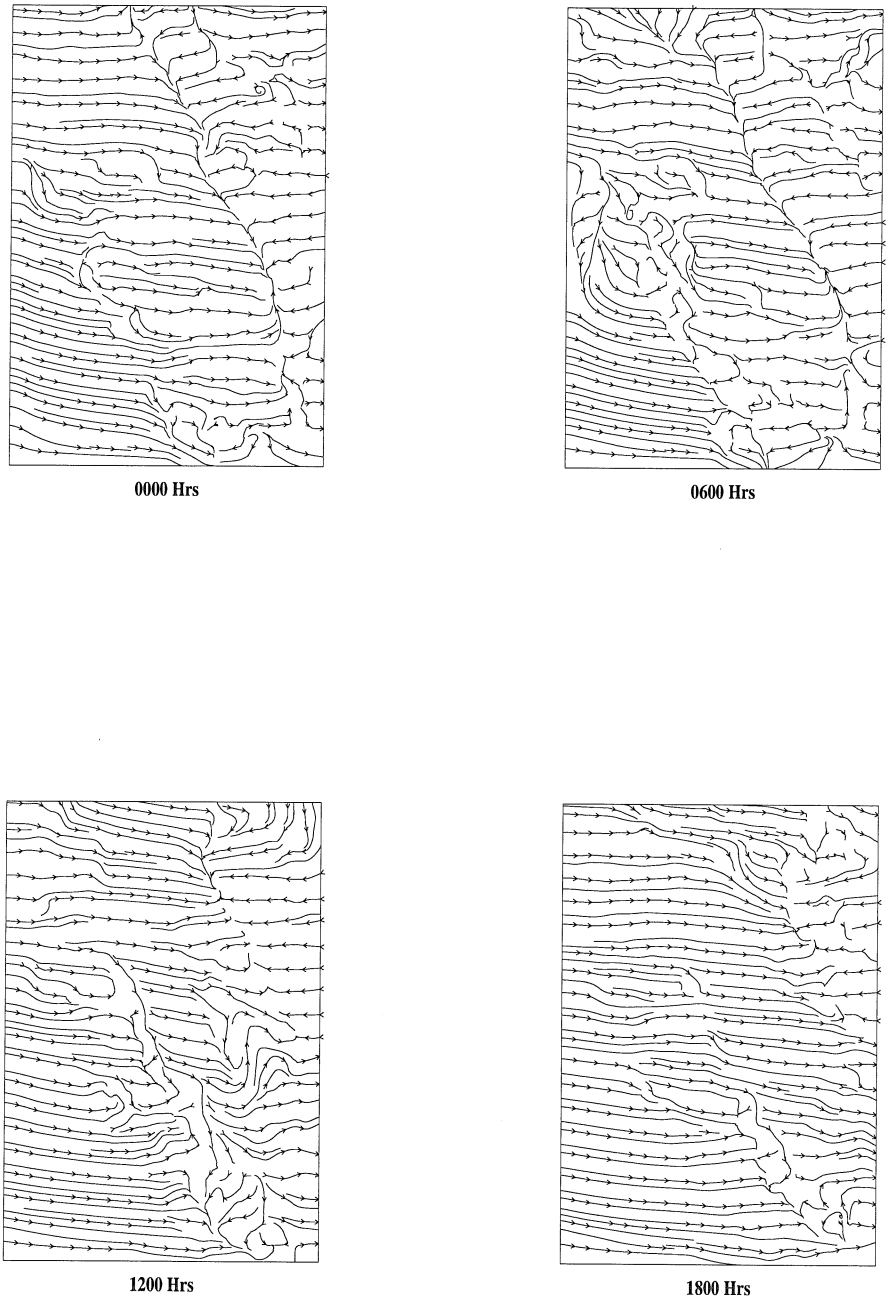


Figure 5.13: Streamlines of the horizontal advective flow field used in the bottom vertical layer for August 5, 1990

The Mercator projection is used for low latitude domains.

$$X = R\cos\varphi_1(\lambda - \lambda_0) \tag{5.47}$$

$$Y = R\cos\varphi_1 \ln((1+\sin\varphi)/\cos\varphi). \tag{5.48}$$

The boundary concentrations used in SARMAP are time independent. Raw data to compute the boundary conditions was obtained from measurements obtained by Blumenthal (1993). Boundary conditions are used to calculate the horizontal flux as described in Equation (5.6). Table 5.5 shows the boundary concentrations used in the August 3-5, 1990 episode at the bottom cells of the modeling region. Figure 5.18 shows the vertical profile of boundary concentrations for all species. Figures 5.14 - 5.17 show CH4, CO, NO, and NO2 concentrations respectively measured during the August 3-5 1990 episode in the San Joaquin Valley.

5.5 Simulation of San Joaquin Valley Episode

Sensitivity analysis is the process of studying, through numerical simulation, the influence on model predictions of variations in one or more model inputs. This section discusses four of the sensitivity analysis performed: zero initial conditions, zero boundary conditions, zero emissions, and zero dry deposition.

5.5.1 Zero Initial Conditions

This run consists of setting the concentration of all species to zero in all cells of the modeling domain at the start of the simulation. Comparison of zero initial conditions predictions with those of the base case reveal the significance of the initial state of the model on especially the second and third day of predicted ozone concentration fields. The closer the predictions of the zero initial conditions run are to those of the base case, the greater the effect of emissions during the episode being modeled. One would expect little residual effects from the initial conditions when the modeling

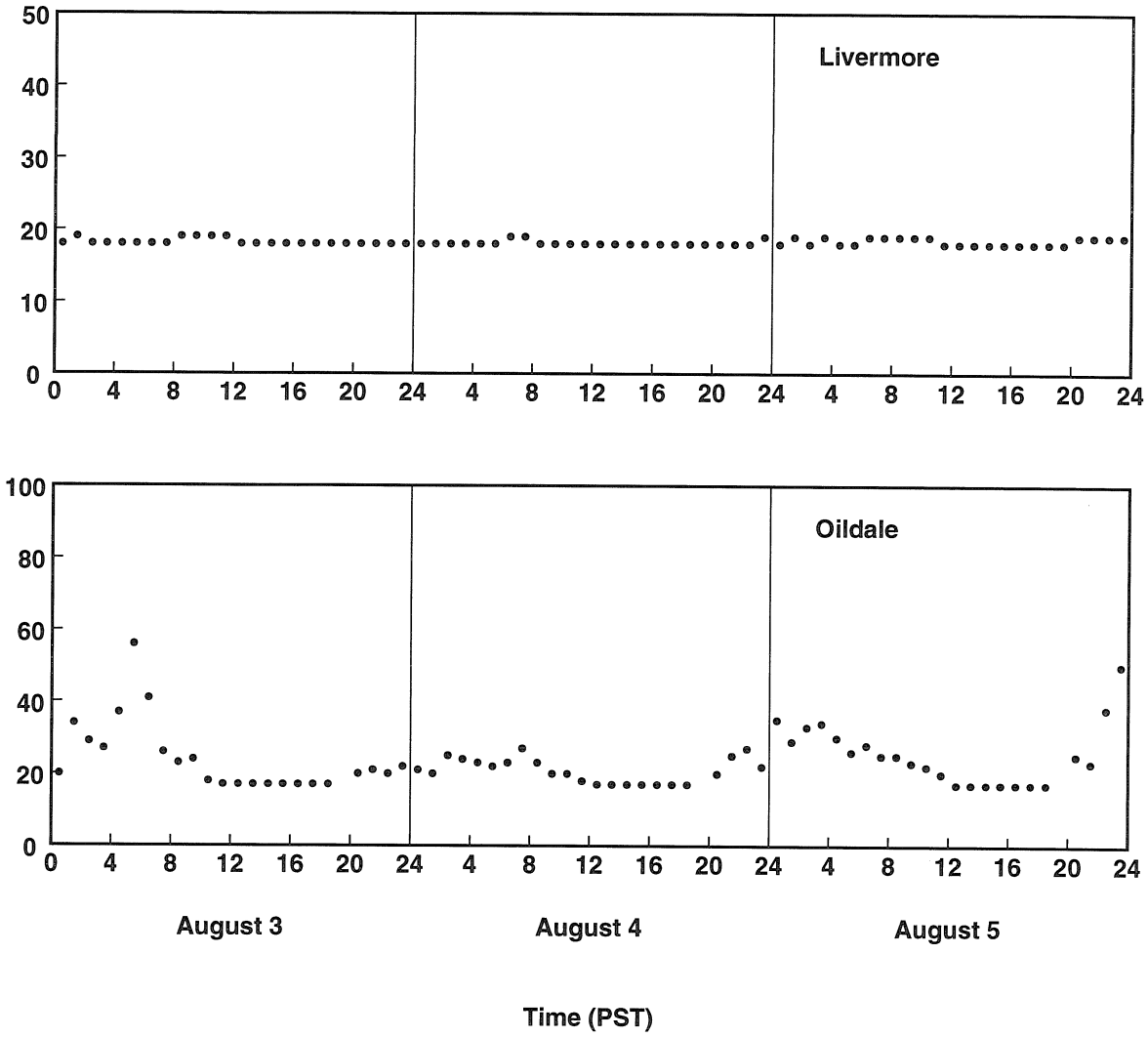


Figure 5.14: Time series plot of observed CH4 concentrations in ppb during the August 3-5, 1990 episode in the San Joaquin Valley of California.

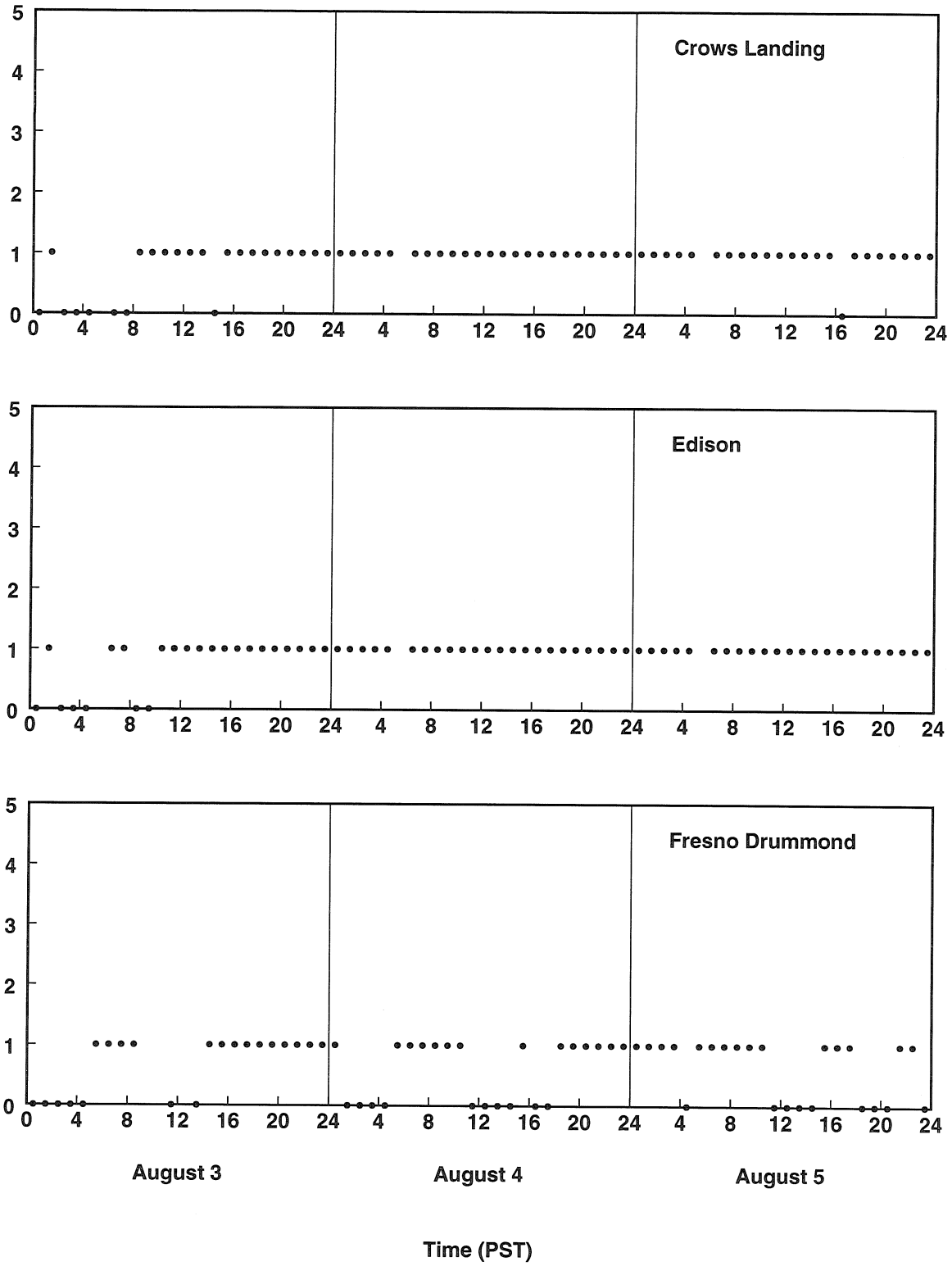


Figure 5.15: Time series plot of observed CO concentrations in ppm during the August 3-5, 1990 episode in the San Joaquin Valley of California.

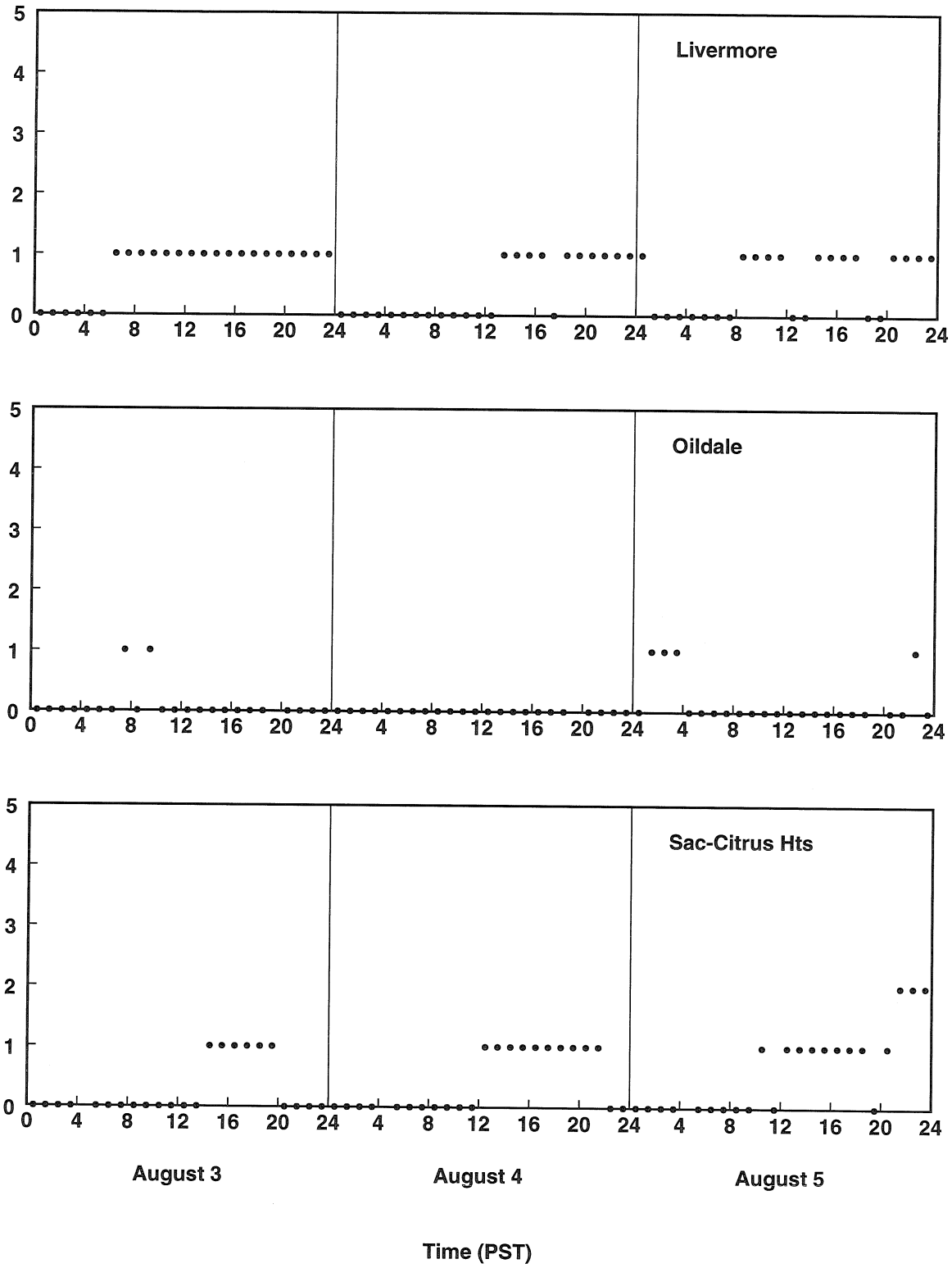


Figure 5.15: Continuation

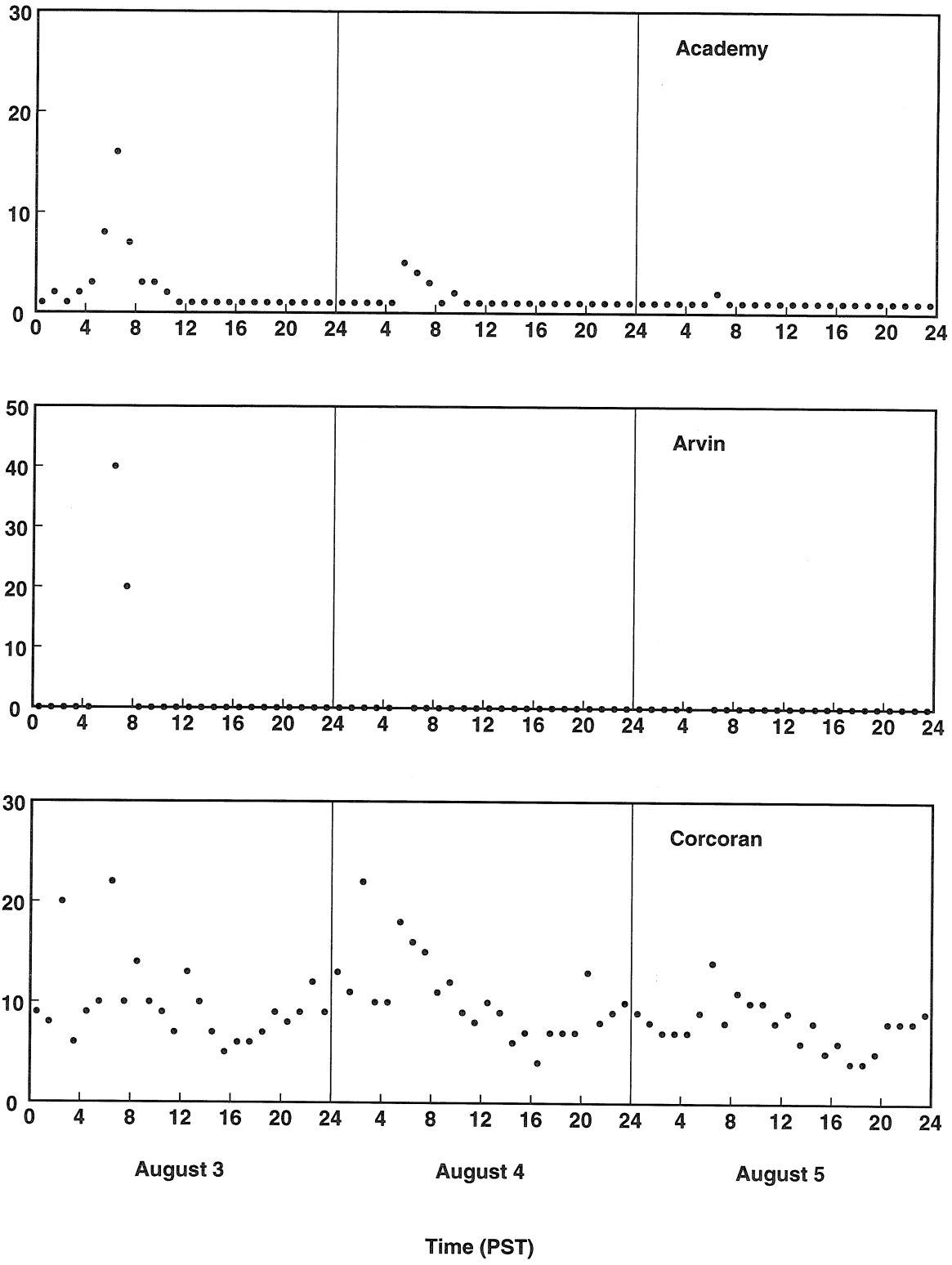


Figure 5.16: Time series plot of observed NO concentrations in ppb during the August 3-5, 1990 episode in the San Joaquin Valley of California.

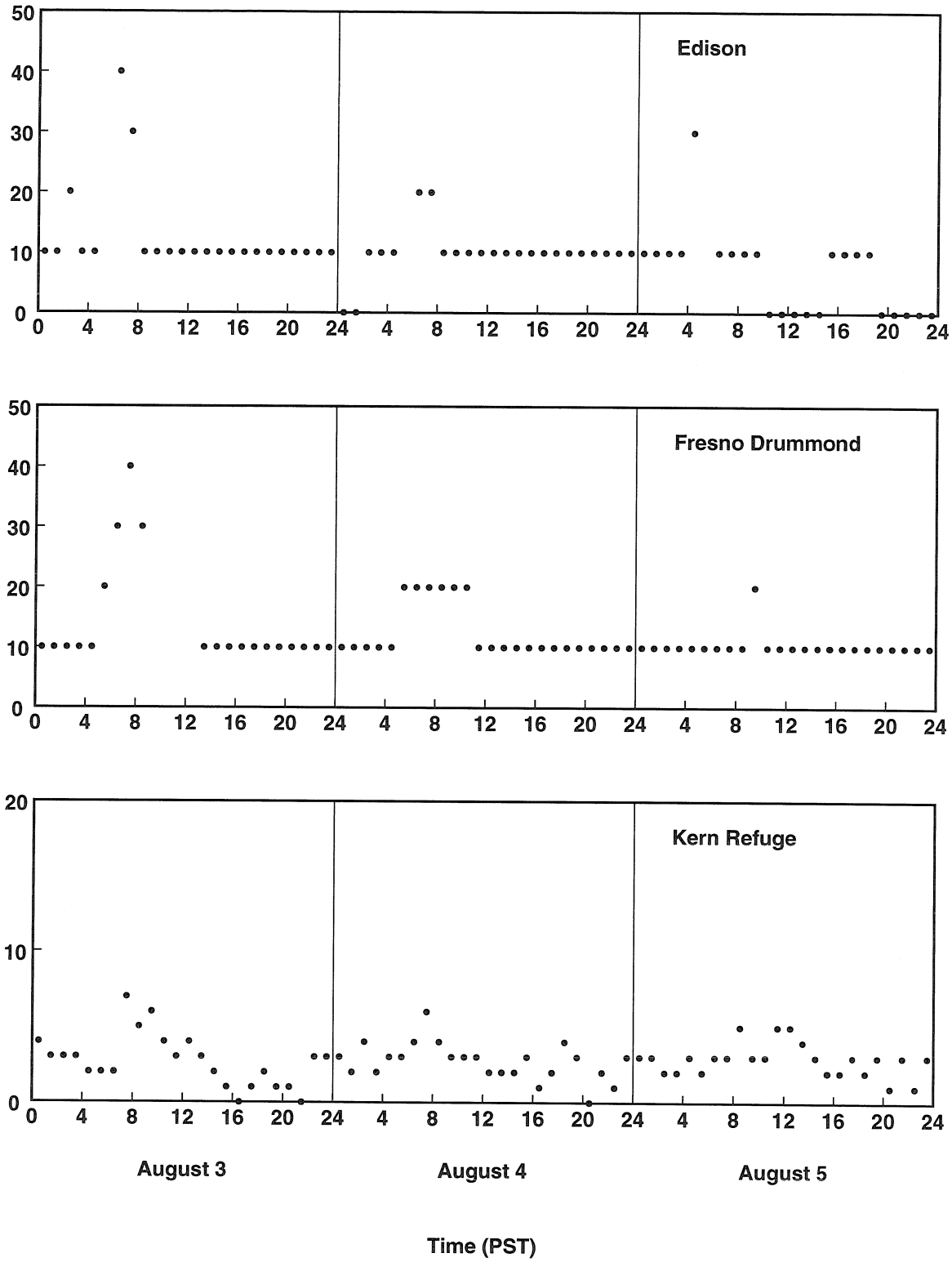


Figure 5.16: Continuation

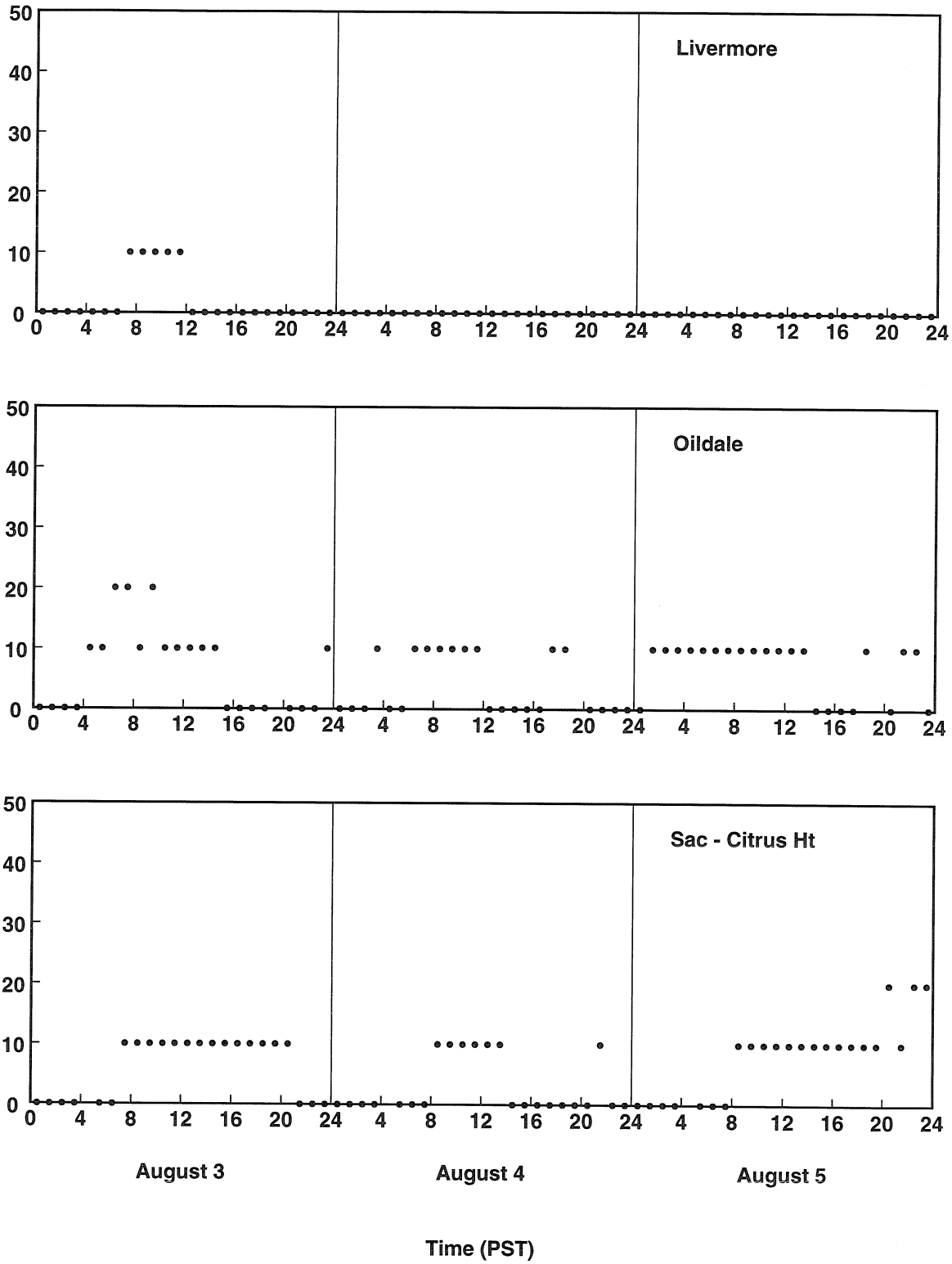


Figure 5.16: Continuation

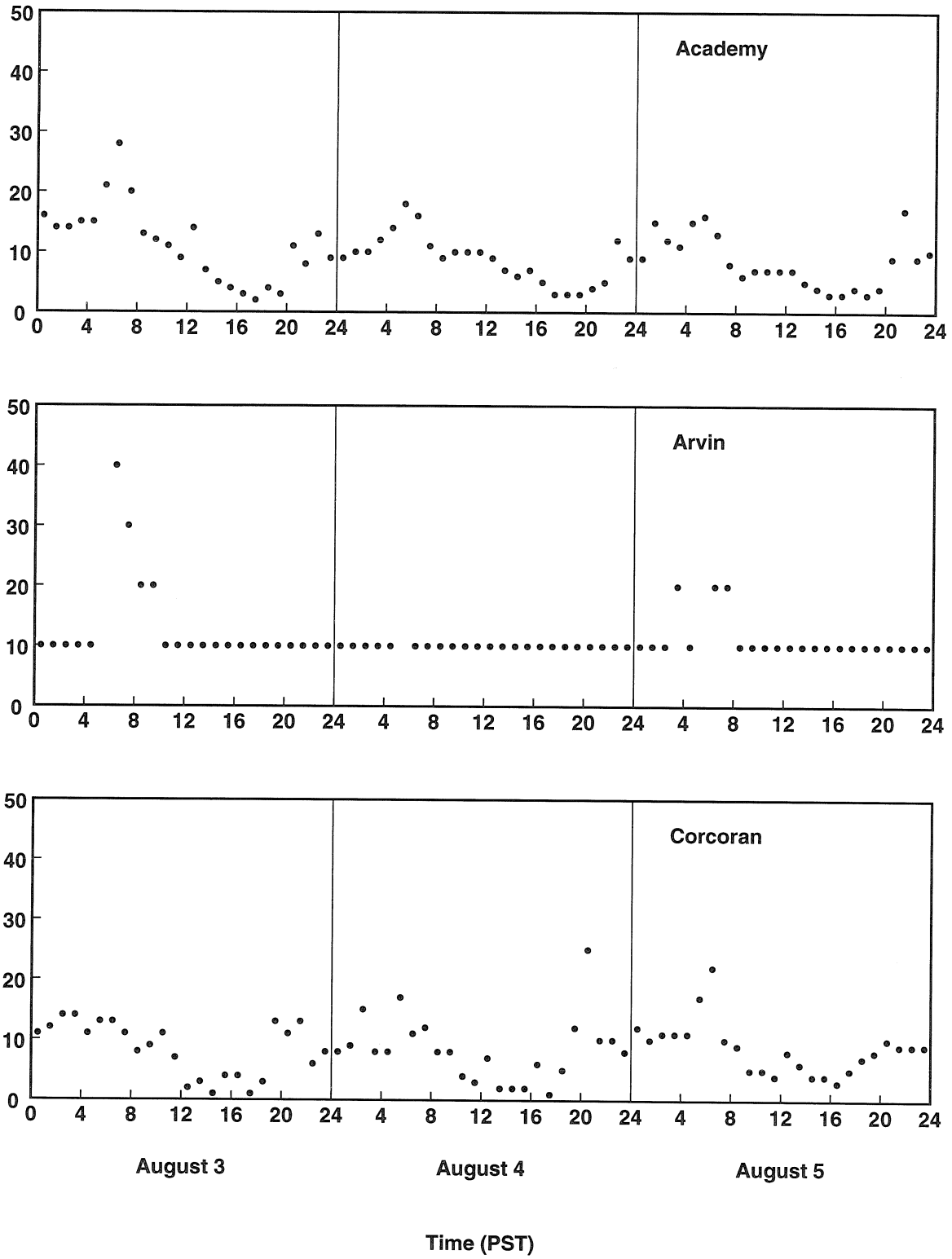


Figure 5.17: Time series plot of observed NO2 concentrations in ppb during the August 3-5, 1990 episode in the San Joaquin Valley of California.

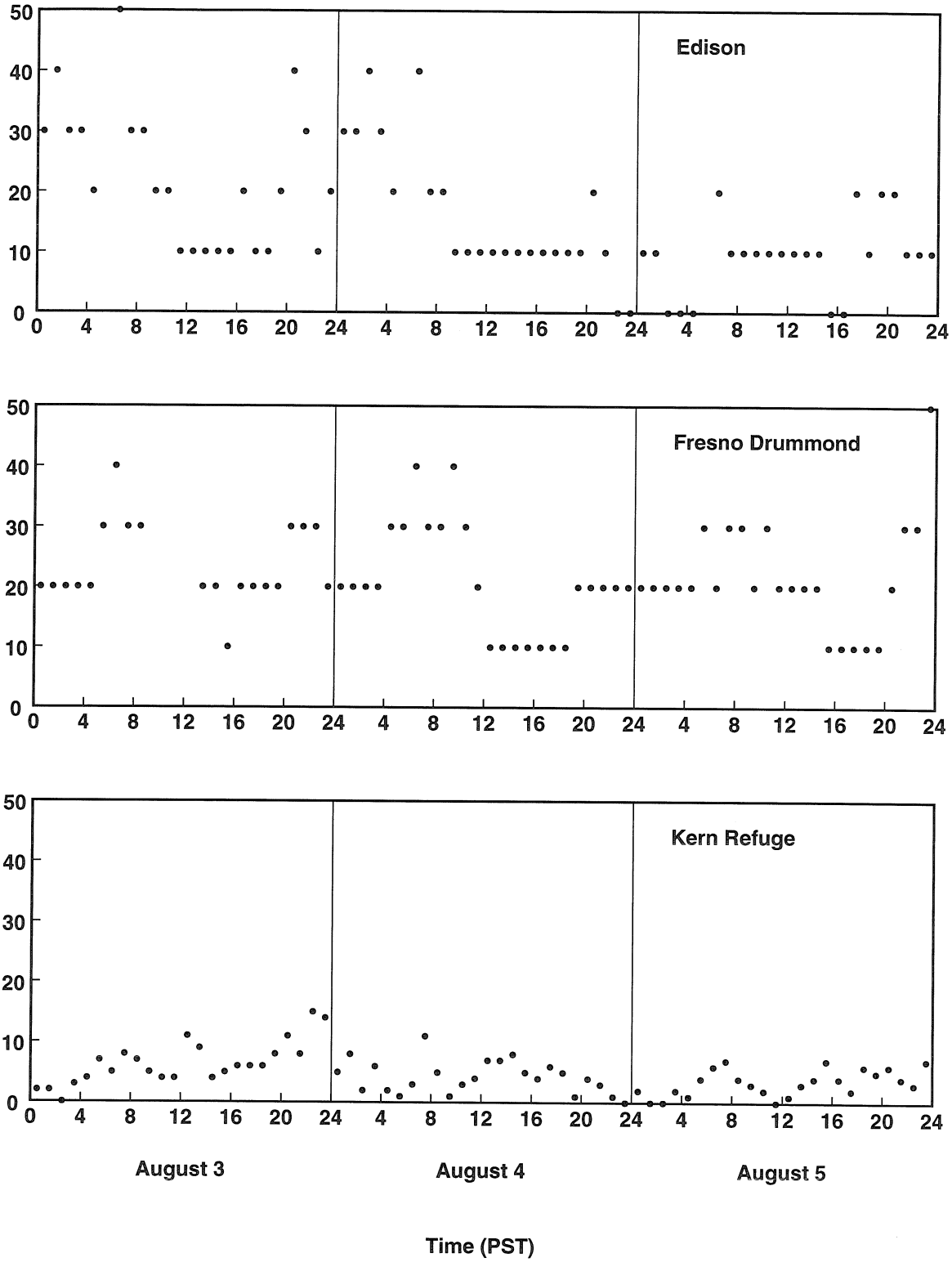


Figure 5.17: Continuation

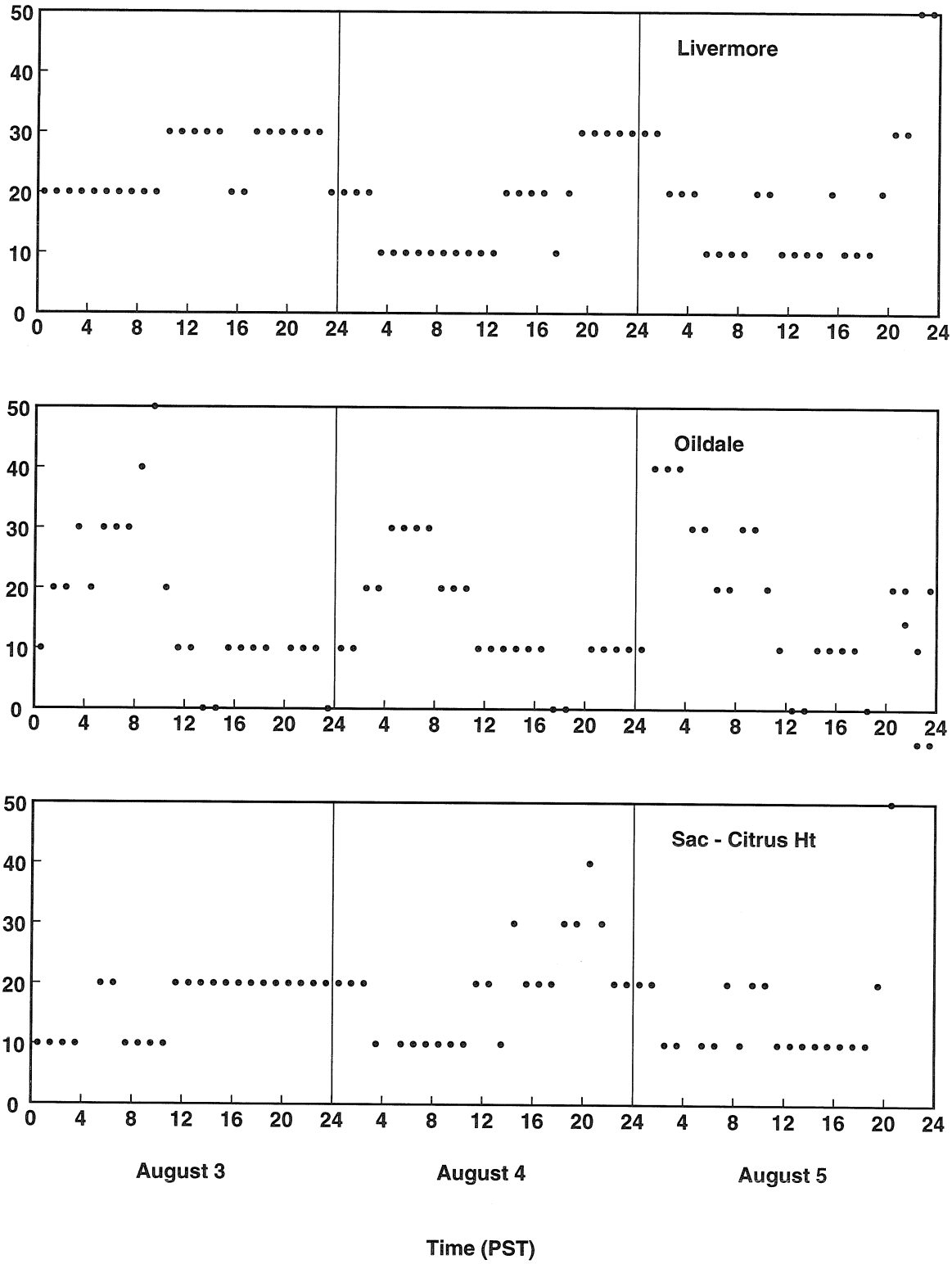


Figure 5.17: Continuation

Table 5.5: Lateral boundary concentrations used in the August 3-5, 1990 episode.

Species code	Concentration (ppb)
SO2	10.000
SULF	0.100
NO2	2.500
NO	0.500
O3	40.000
HNO3	0.010
H2O2	0.010
ALD	1.524
HCHO	5.784
OPEN	0.010
PAR	41.100
ETH	1.404
OLE	0.828
ISO	0.100
NH3	0.10E-06
N2O5	0.100
NO3	0.100
PAN	0.100
TOL	0.492
XYL	0.270
CRES	0.100
MGLY	0.100
CO	200.000
C2O3	0.10E-06
HONO	0.100
HNO4	0.100
CH4	1700.000
HO	0.10E-06
HO2	0.10E-06

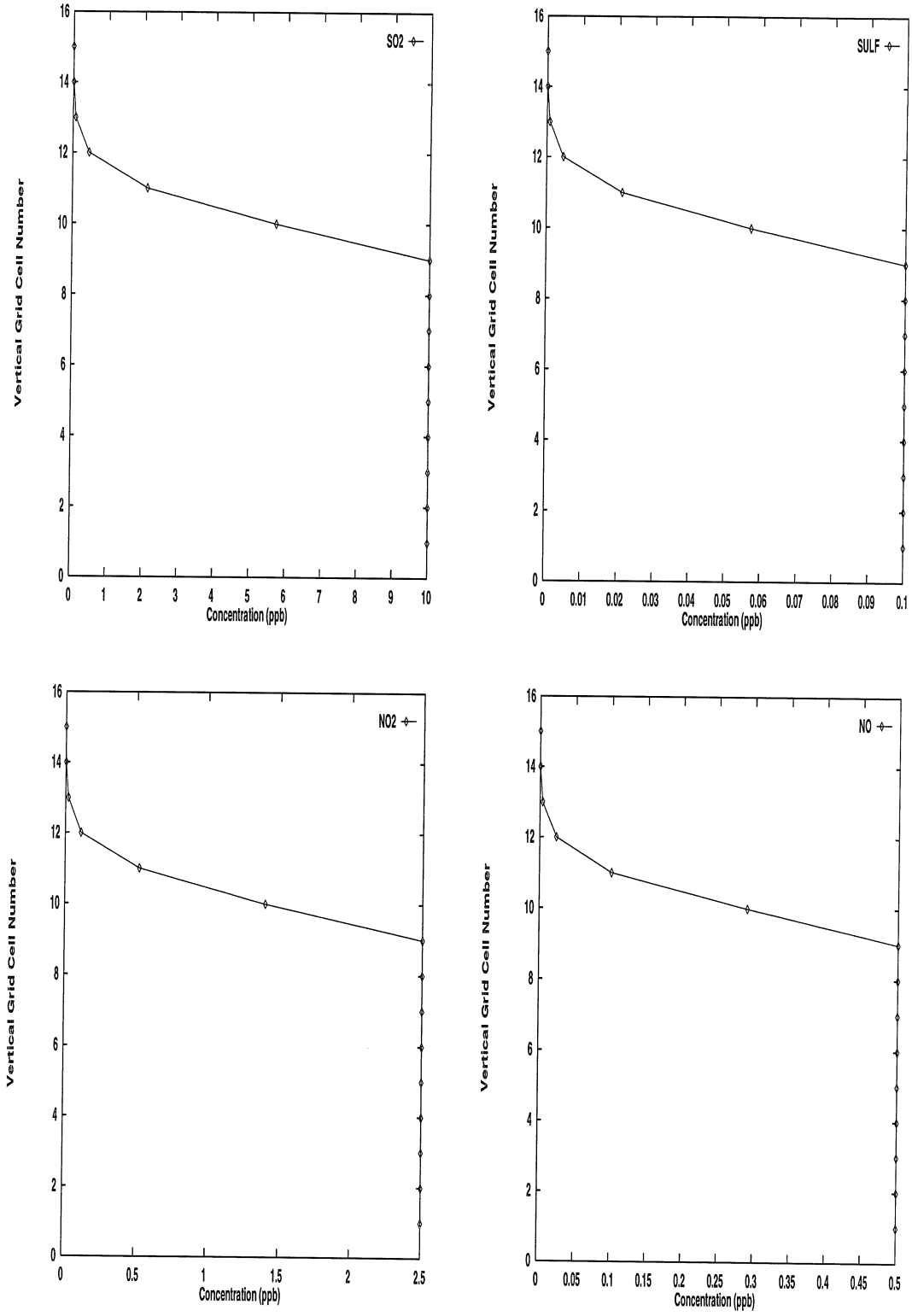


Figure 5.18: Lateral boundary concentrations for all species used in the August 3-5, 1990 episode of the San Joaquin Valley of California.

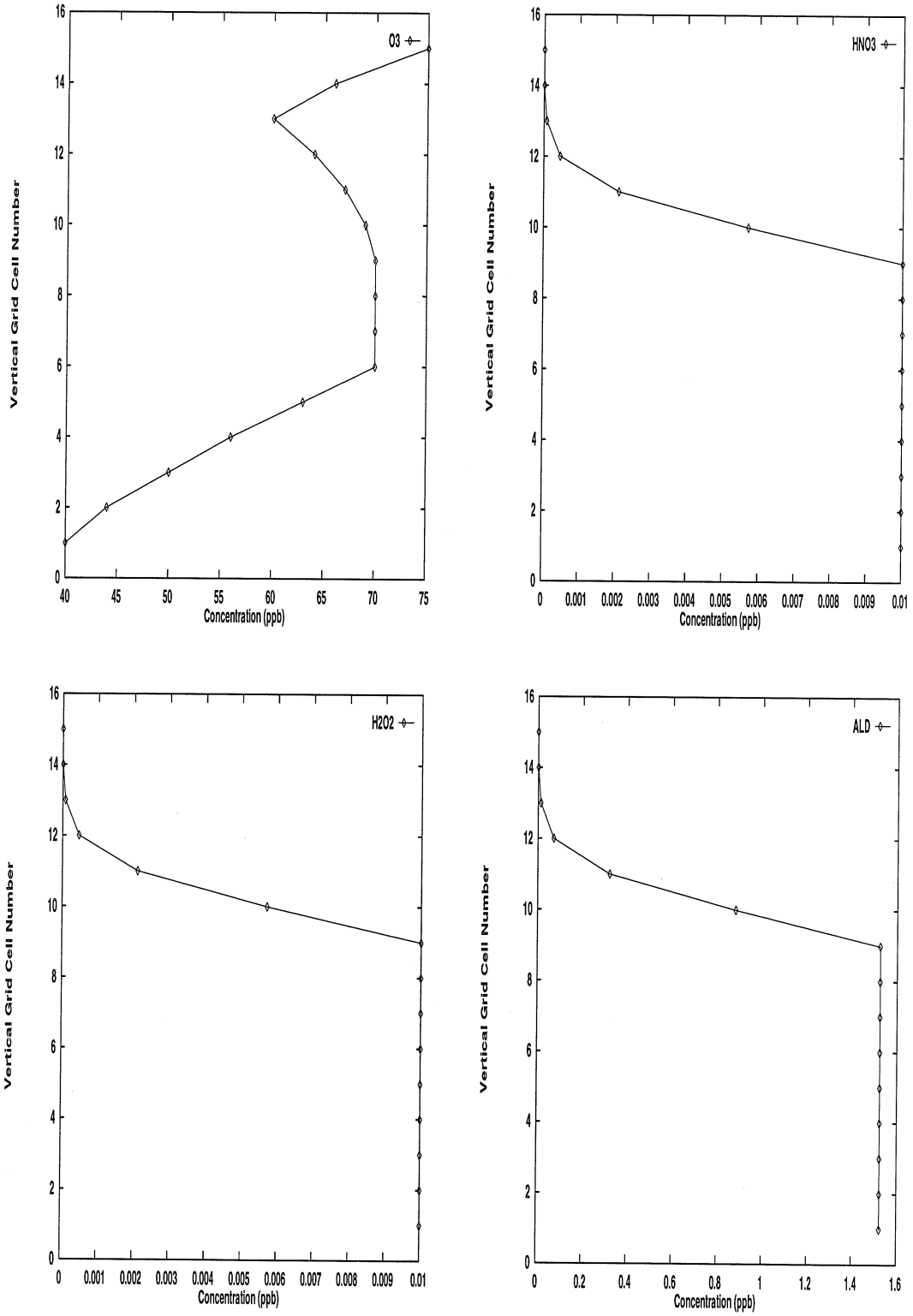


Figure 5.18: (Continuation)

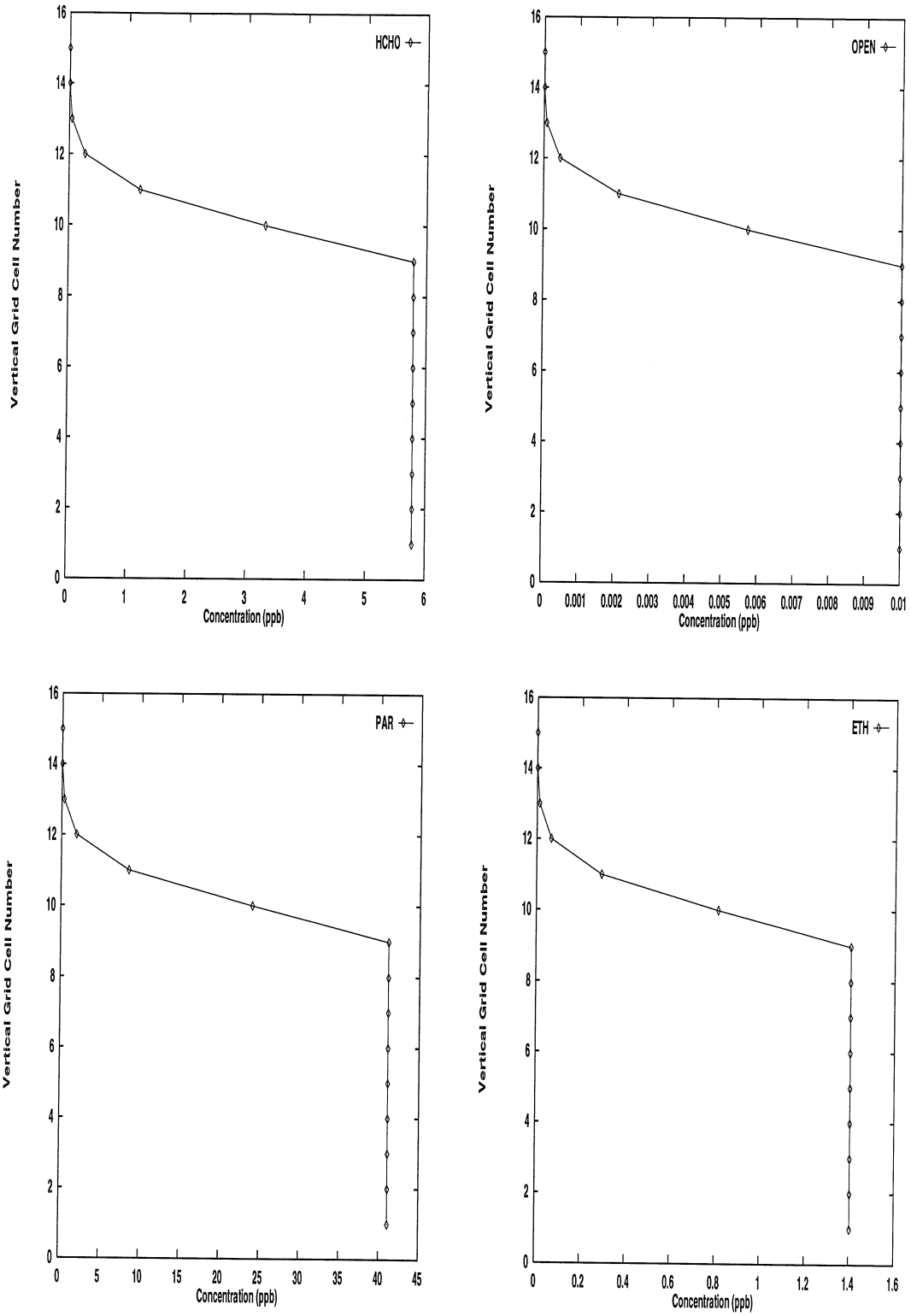


Figure 5.18: (Continuation)

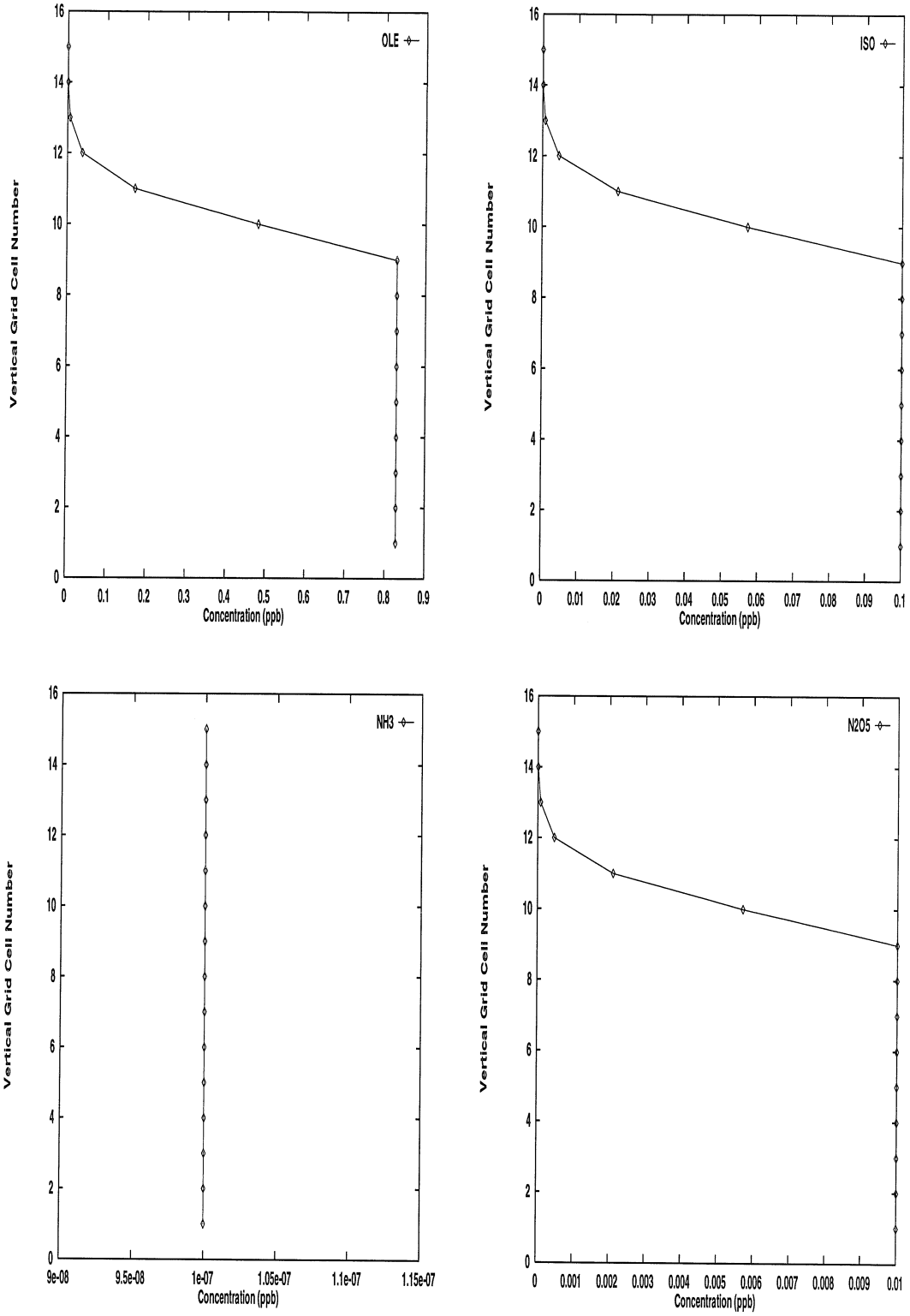


Figure 5.18: (Continuation)

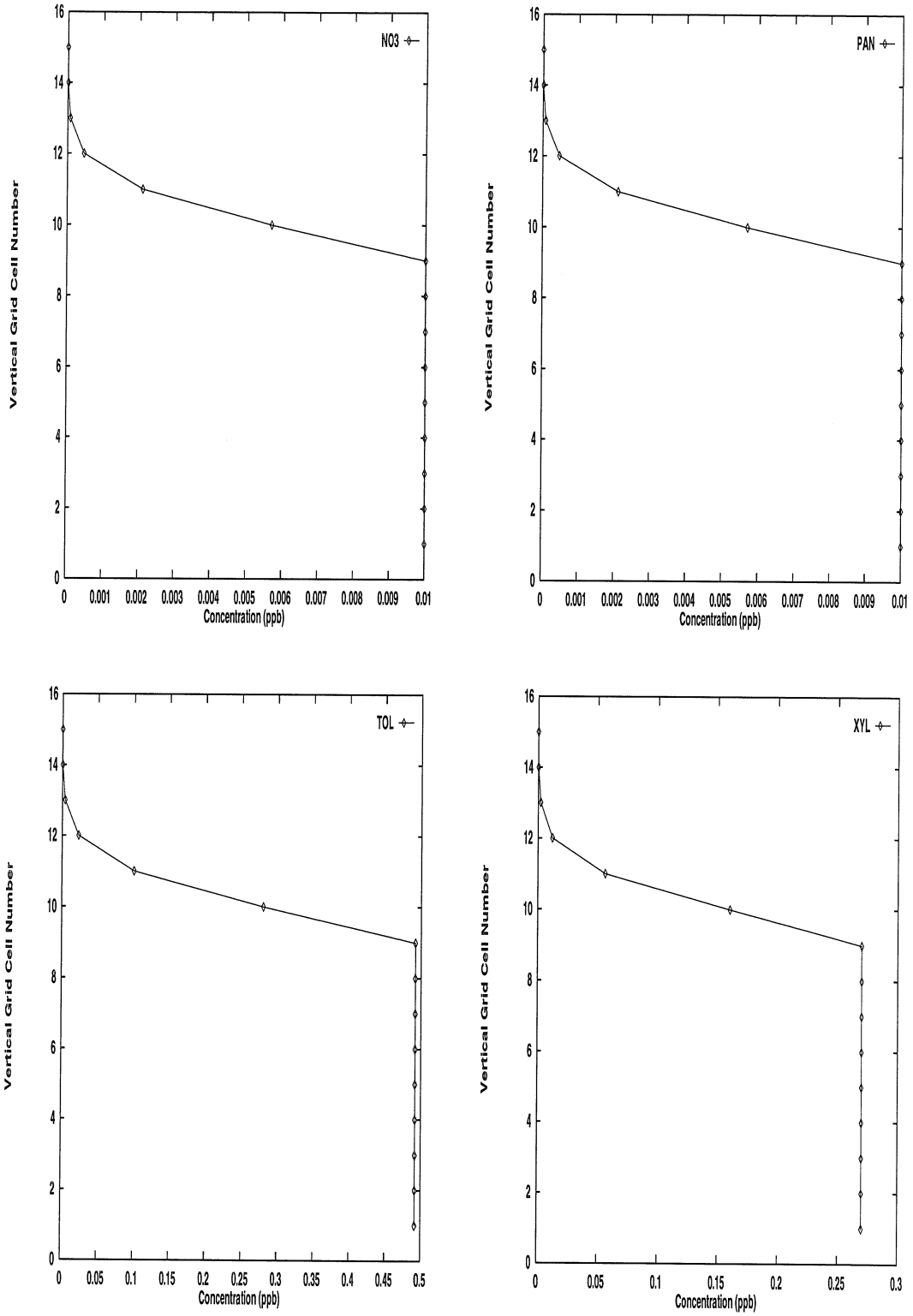


Figure 5.18: (Continuation)

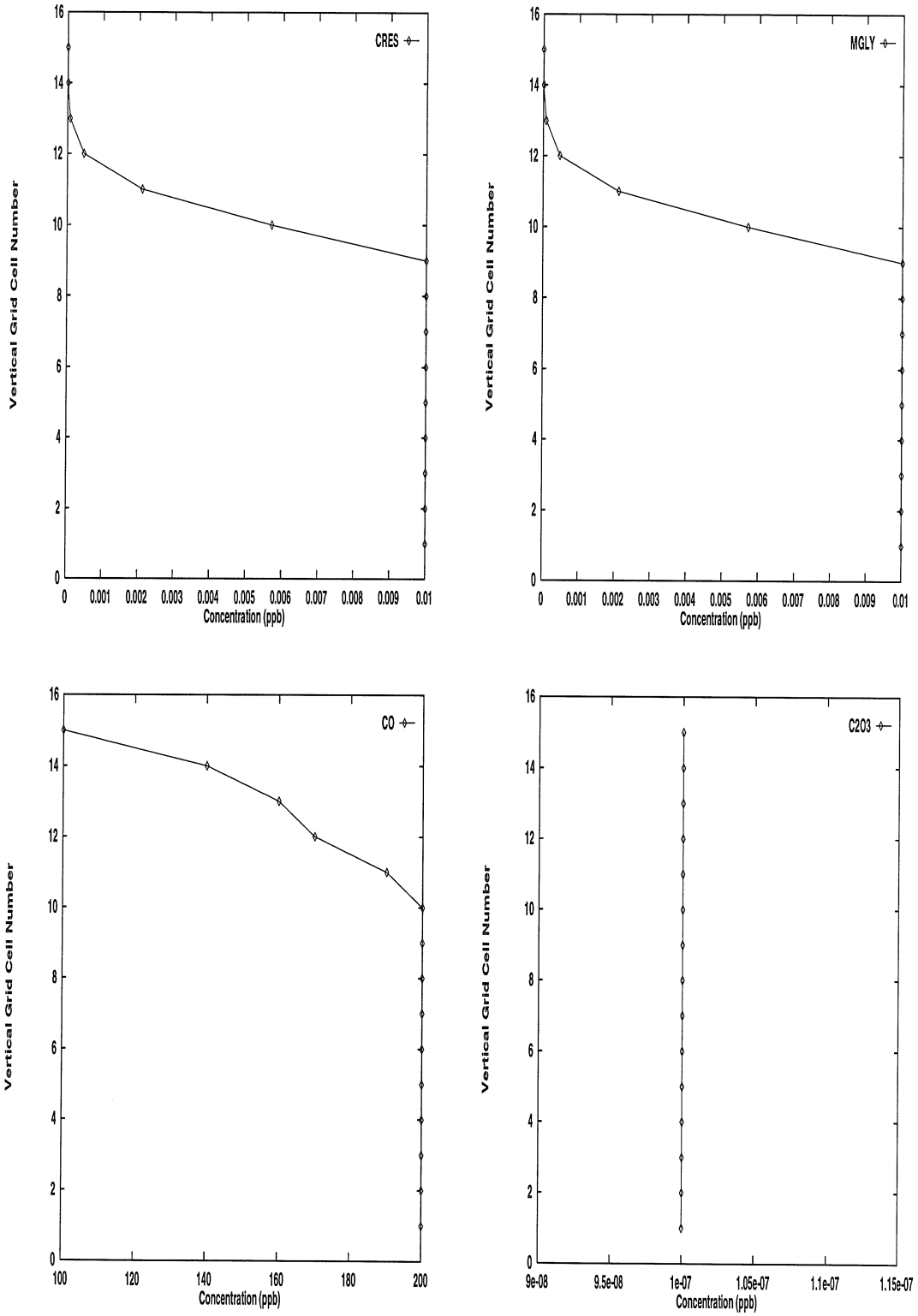


Figure 5.18: (Continuation)

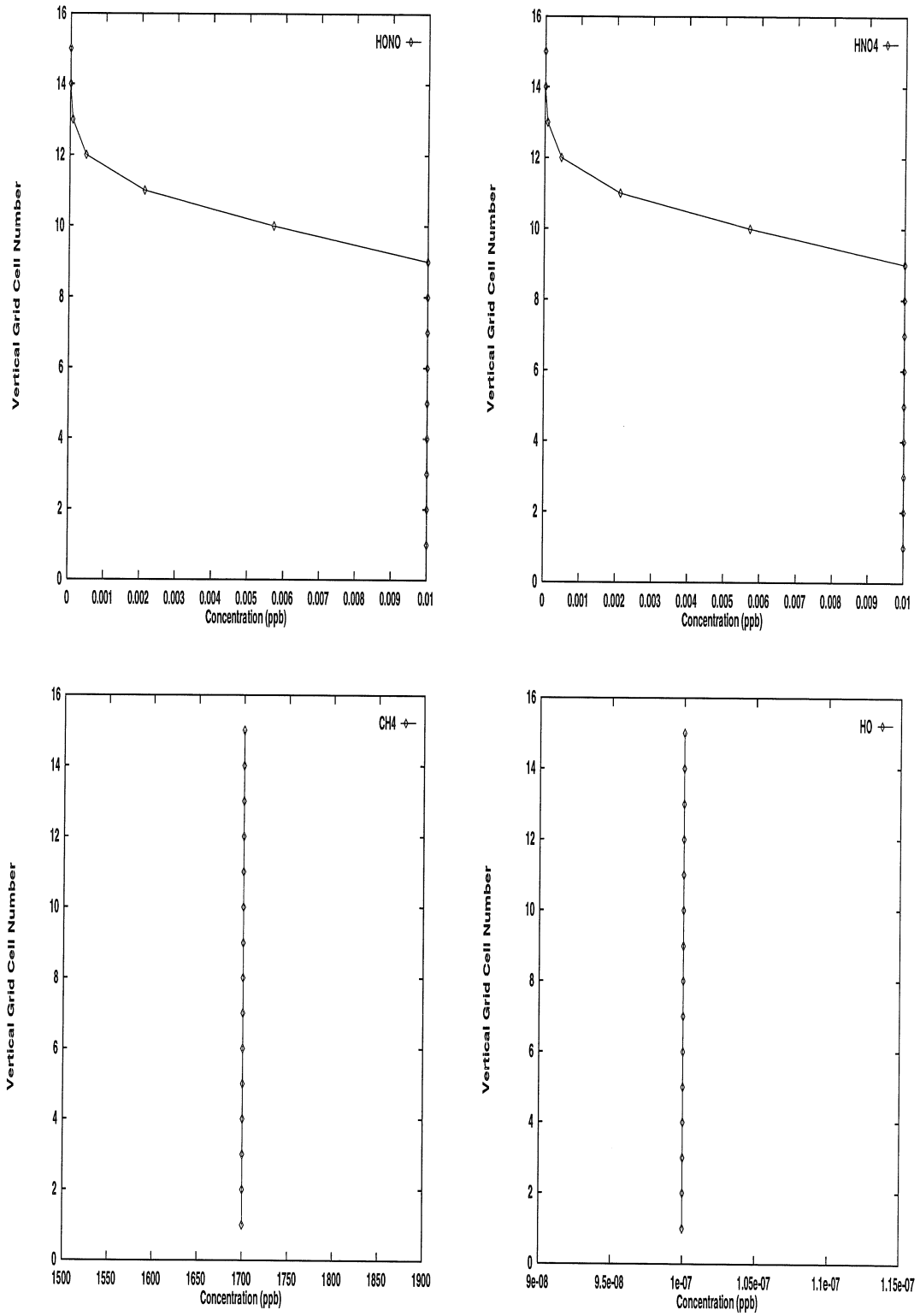


Figure 5.18: (Continuation)

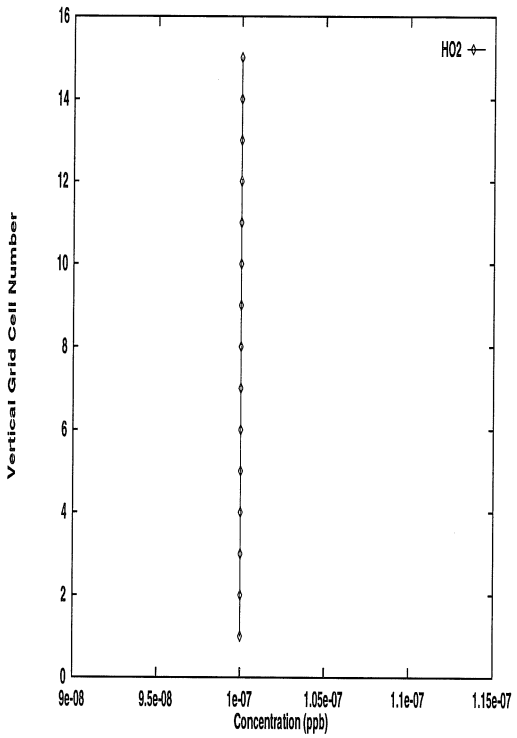


Figure 5.18: (Continuation)

domain is sufficiently flushed out.

Figure 5.19 compares the results from the zero initial condition run with the base case run for the August 1990 episode. The simulation started at 0500 hrs on August 2, 1990. It is observed that as early as August 3 there are virtually no discrepancies between the zero initial conditions predictions and the base case predictions on Sacramento, Stockton, Livermore, Crows Landing, and Gilroy. The dynamics of ozone production at these sites are dominated by boundary conditions and emissions. As shown in Figures 5.7 - 5.9 these sites are downwind from high emission locations and near inflow boundaries. On the other hand, locations like Academy and Edison exhibit greater sensitivity to initial conditions. Even though, Edison is close to the edge of the computational domain, the pollutant dynamics of the site are not dominated by boundary conditions since it is located at an outflow boundary. Downwind sites start to show small discrepancies on the middle of the second day of the simulation. Results indicate that ozone concentrations on the third day of the simulation at all sites are practically identical to those produced by the base case run; the expected behavior of a non-stagnant episode.

5.5.2 Zero Boundary Conditions

The zero boundary condition simulation consists of setting all species concentrations at all horizontal boundaries to zero during all times of the episode. The comparison of the zero boundary condition run with that of the base case run allows one to determine the dominance of boundary transport on the predictions. The greater the discrepancy between the two predictions the greater the effect the boundary conditions have on the model.

Figure 5.20 compares the results from the zero boundary conditions run with the base case run for the August 1990 episode. Results show that the diurnal pattern in the ozone concentration for the zero boundary conditions run is maintained. This indicates that it is emissions, not boundary conditions, that are responsible for temporal variations in the ozone levels. However ozone levels in Sacramento, Stockton,

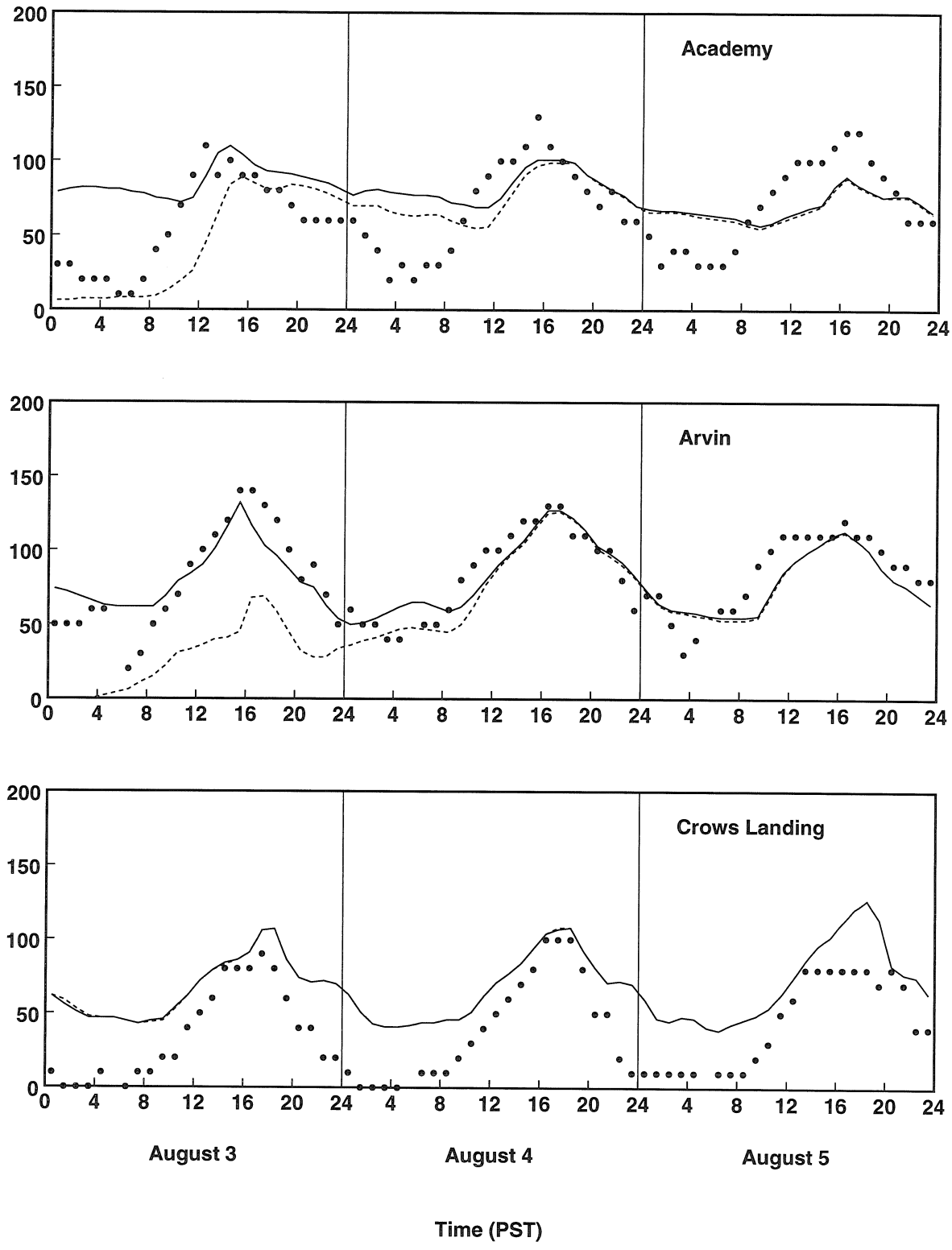


Figure 5.19: Time series plot of observed ozone concentrations in ppb (solid circles) and base case ozone predictions (solid line) and zero initial conditions ozone predictions (dashed line).

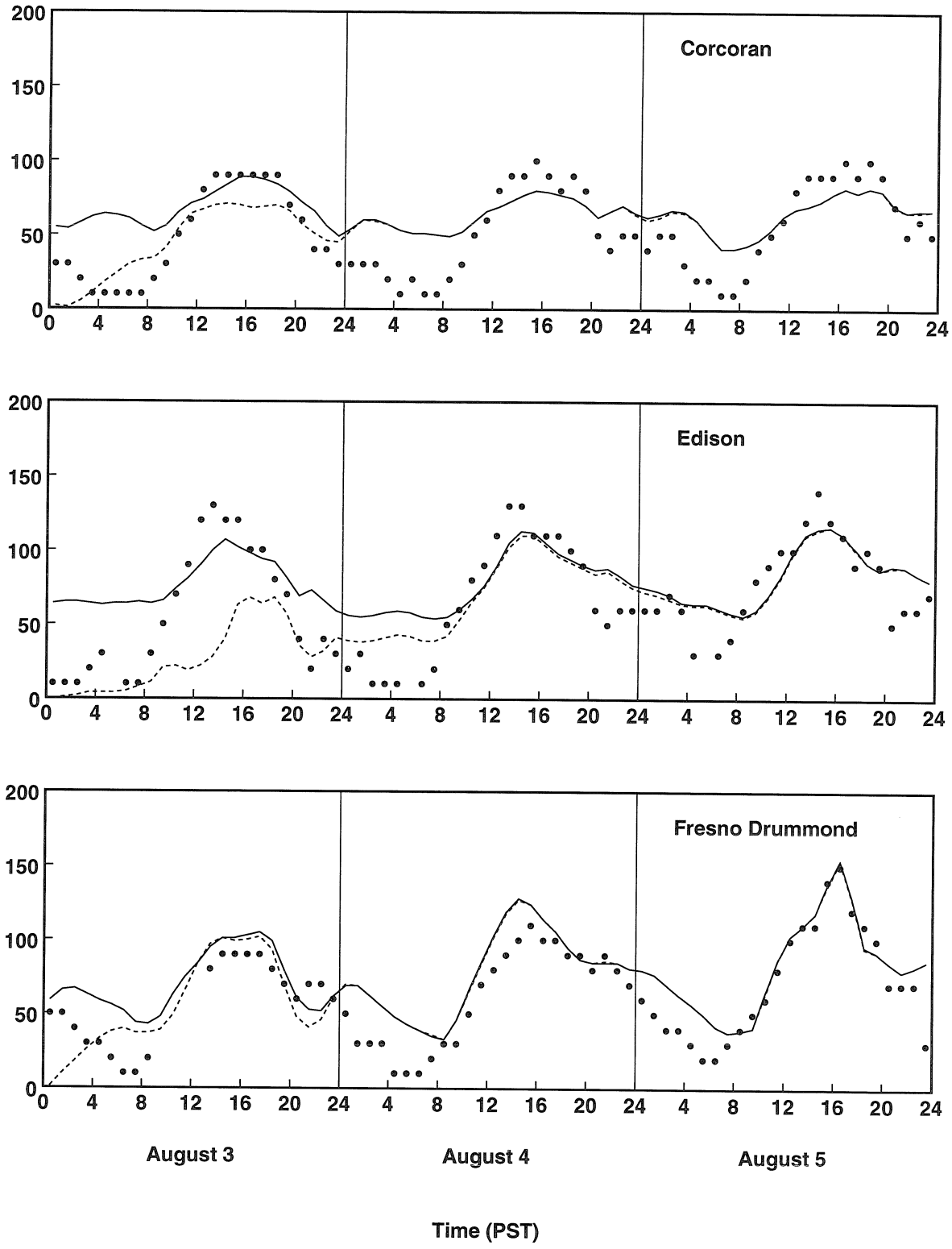


Figure 5.19: (Continued)

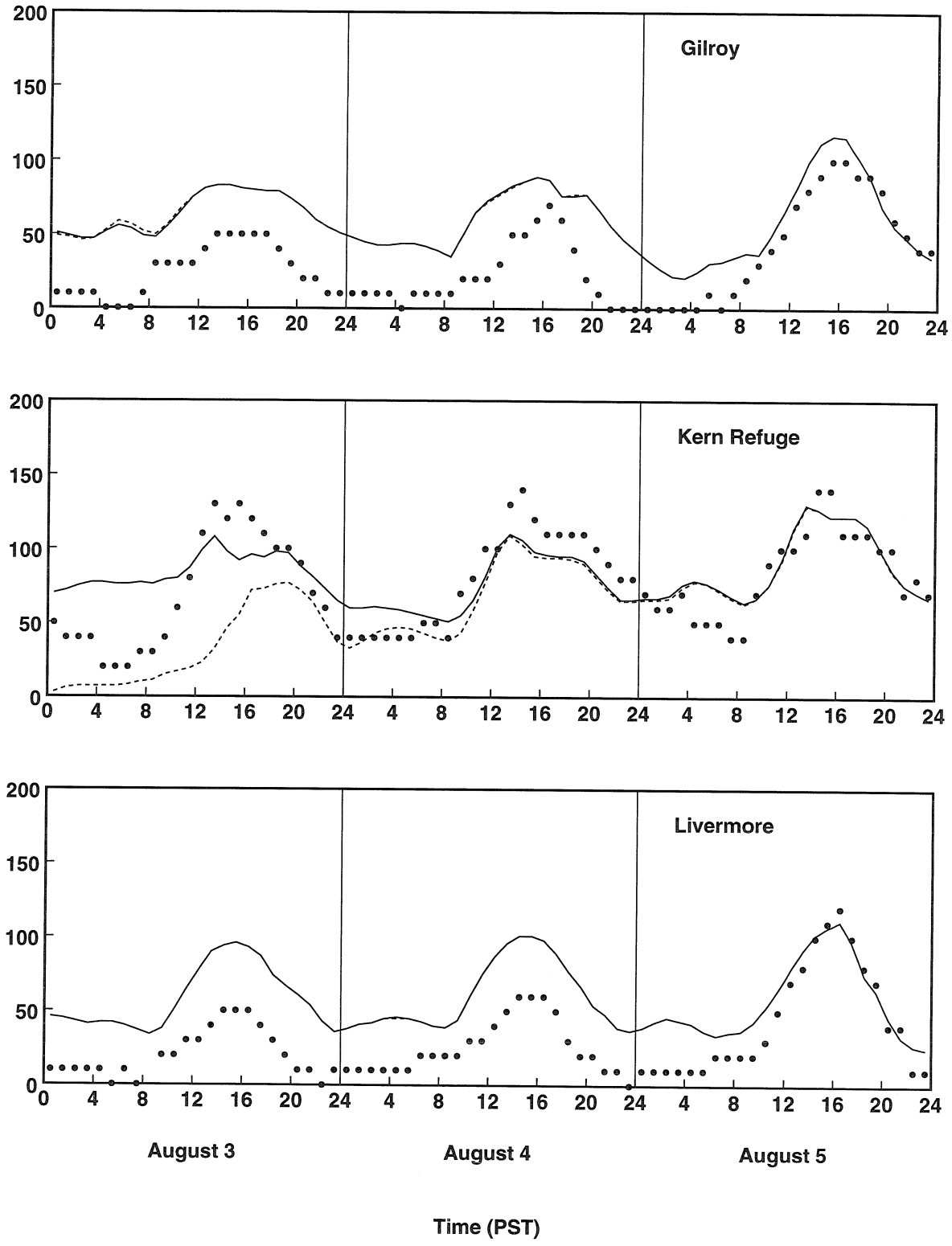


Figure 5.19: (Continued)

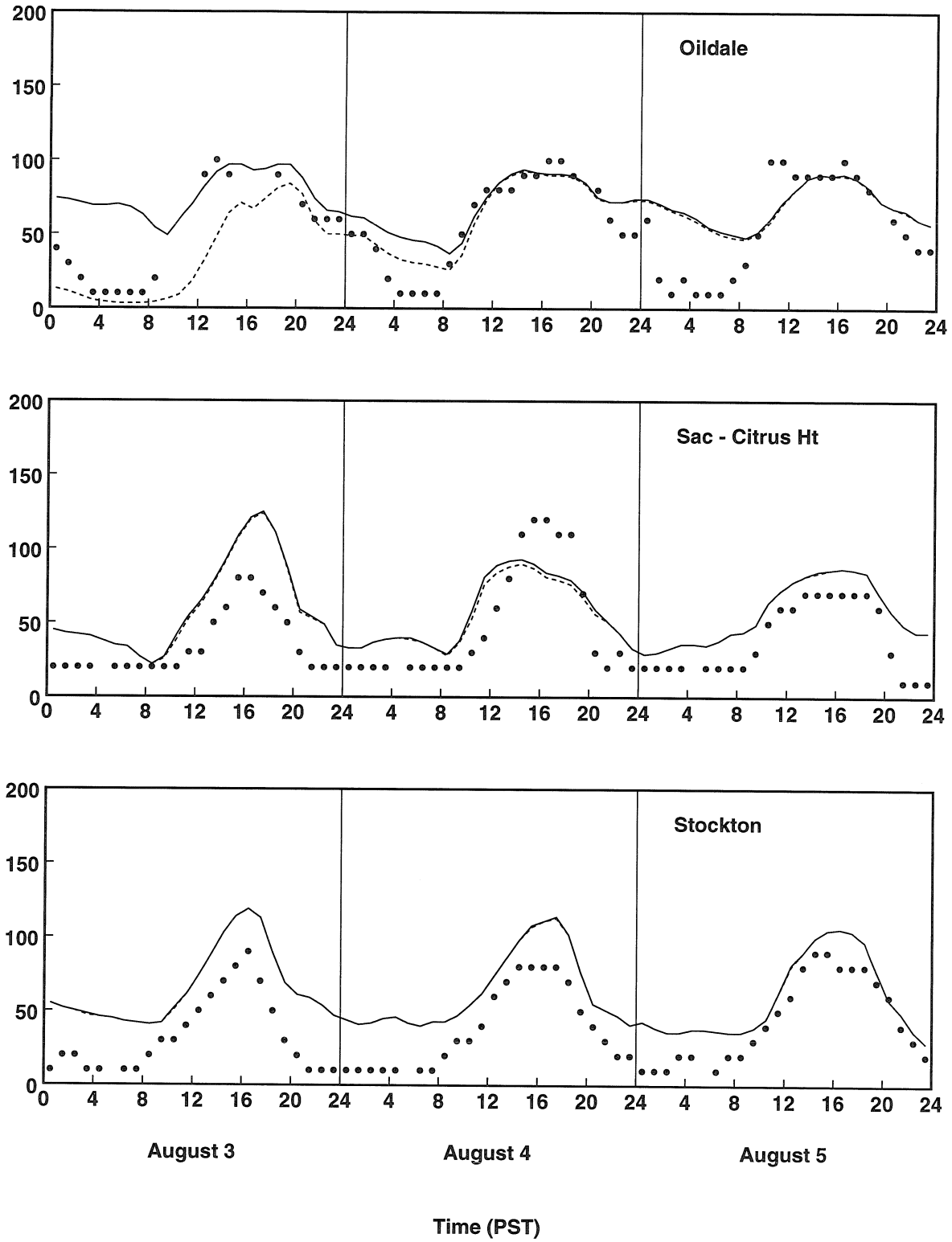


Figure 5.19: (Continued)

Livermore, Crows Landing, and Gilroy exhibit a strong sensitivity to the boundary conditions. Downwind sites such as Kern and Oildale exhibit smaller sensitivity to the boundary conditions. Furthermore, as the simulation time progresses the dependence on boundary conditions on this sites increases. This effect indicates that the parcel of air originally located near the eastern model boundary has been advected through the domain by the second day of the simulation.

The effects of zero boundary conditions on the model are also shown in Figure 5.21, where the the total mass of ozone present in the entire computational domain is plotted as a function of time. The solid line corresponds to the base case run with chemistry and boundary conditions. This line shows the typical ozone peaks that occur in the afternoon. The large-dashed line corresponds to the base case run with the chemistry artificially turned off. This line shows a relatively constant ozone mass. The short-dashed line corresponds to the base case run with chemistry but with zero boundary conditions. As expected the total mass of ozone decreases with time as the polluted air flows out of the computational domain, and is replaced with clean air. Nevertheless, the total mass of ozone is excessively reduced by the model when the boundary conditions are zeroed.

A more typical zero boundary condition model response is shown in Figure 5.22. The results shown in Figure 5.22 correspond to modeling the August 26-28, 1987 episode of the South Coast Air Basin of California using the CIT model. The total ozone mass of the zero boundary condition run is lower than the total ozone mass of the base case run as expected. The CIT model, however, does not show an excessive response to the zeroing of the boundary conditions. Furthermore, Figure 5.22 shows that as time advances the ozone peaks continue to decrease in the zero boundary condition run. This is expected since the polluted air is advected out of the modeling region and replaced with clean air. The total ozone mass for the zero boundary condition case increases in the late afternoon hours since ozone is produced by photochemical activity.

One of the possible reasons that might explain such an excessive reduction in total mass observed in the zero boundary condition run of SARMAP is the fact that

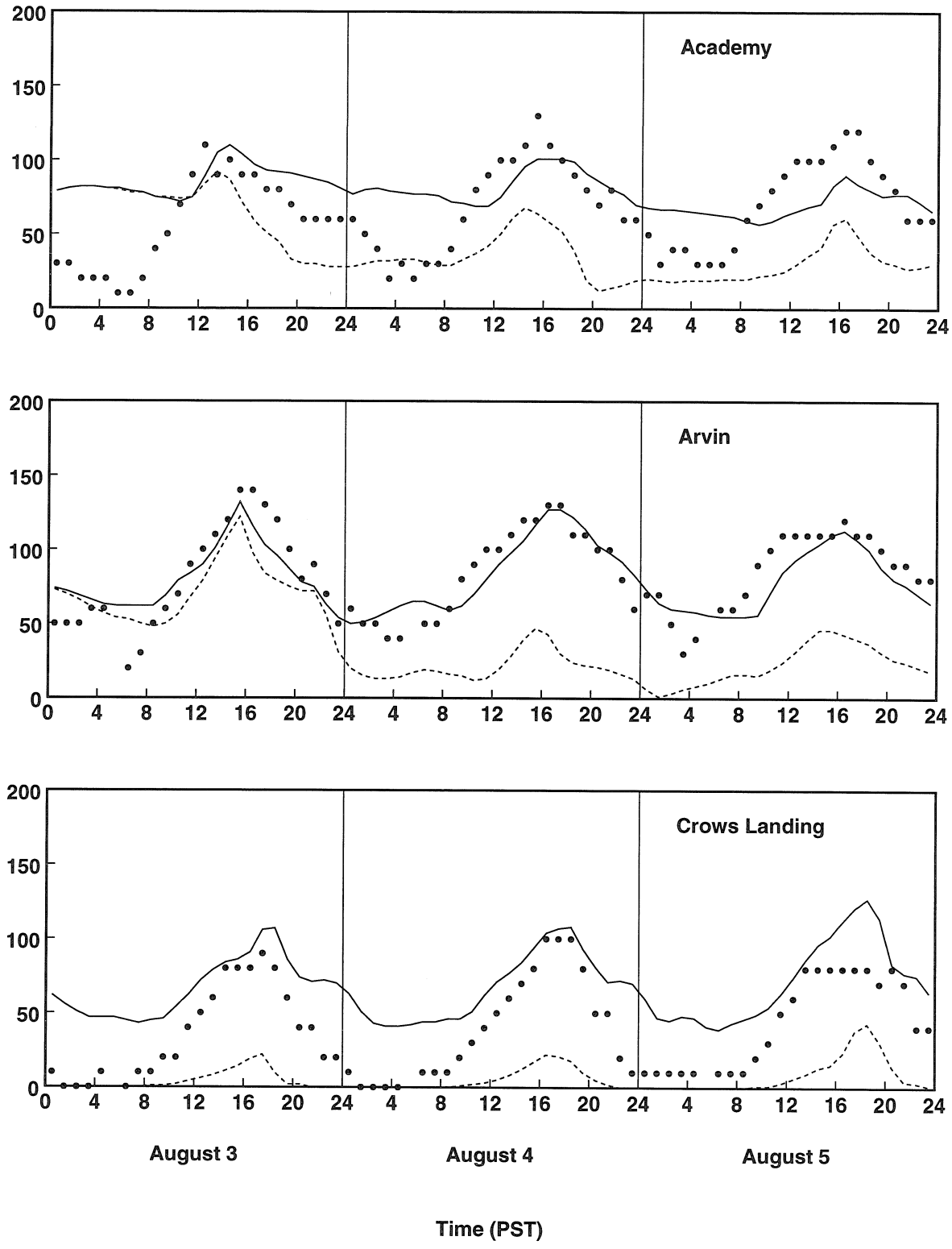


Figure 5.20: Time series plot of observed ozone concentrations in ppb (solid circles) and base case predictions (solid line) and zero boundary conditions ozone predictions (dashed line).

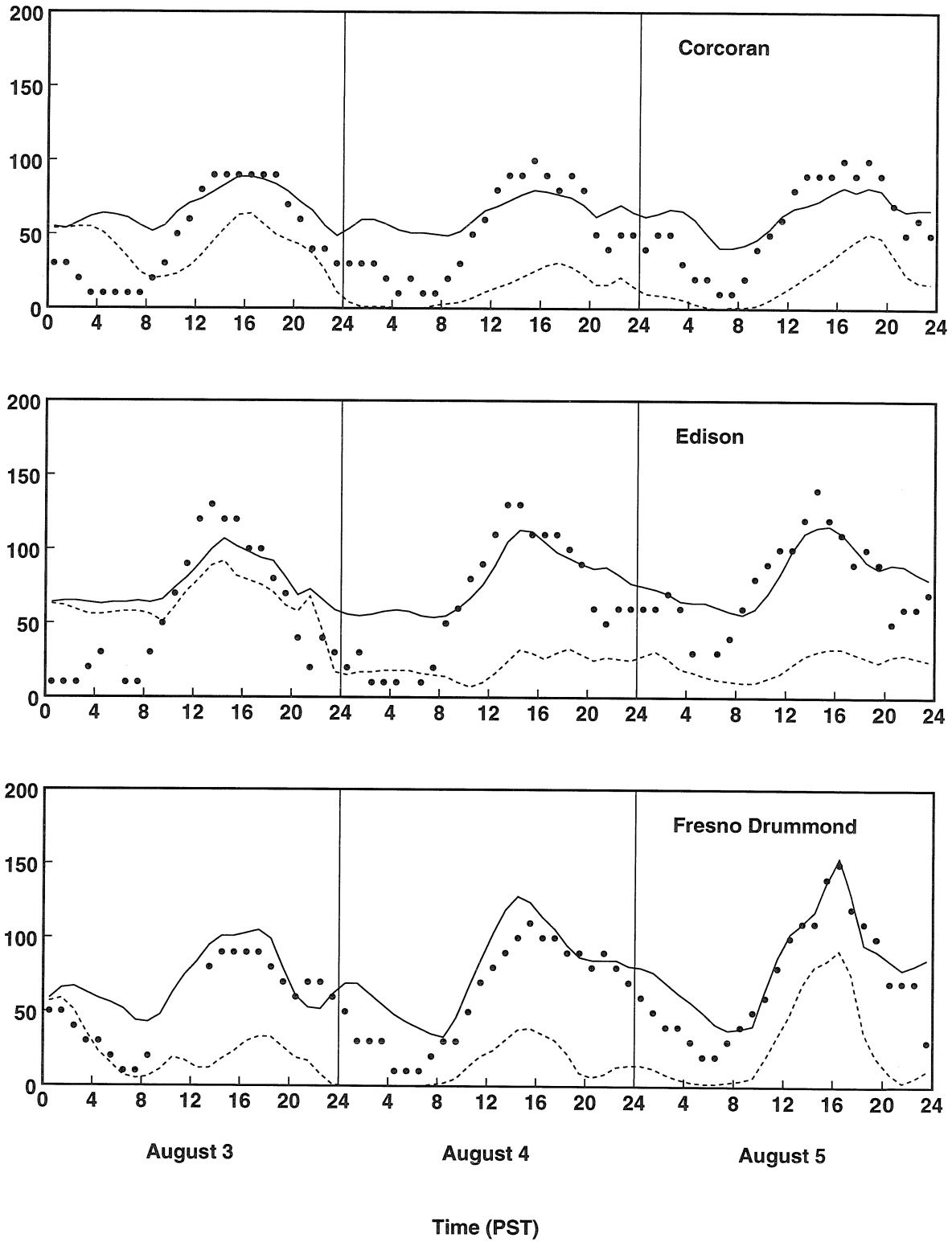


Figure 5.20: (Continued)

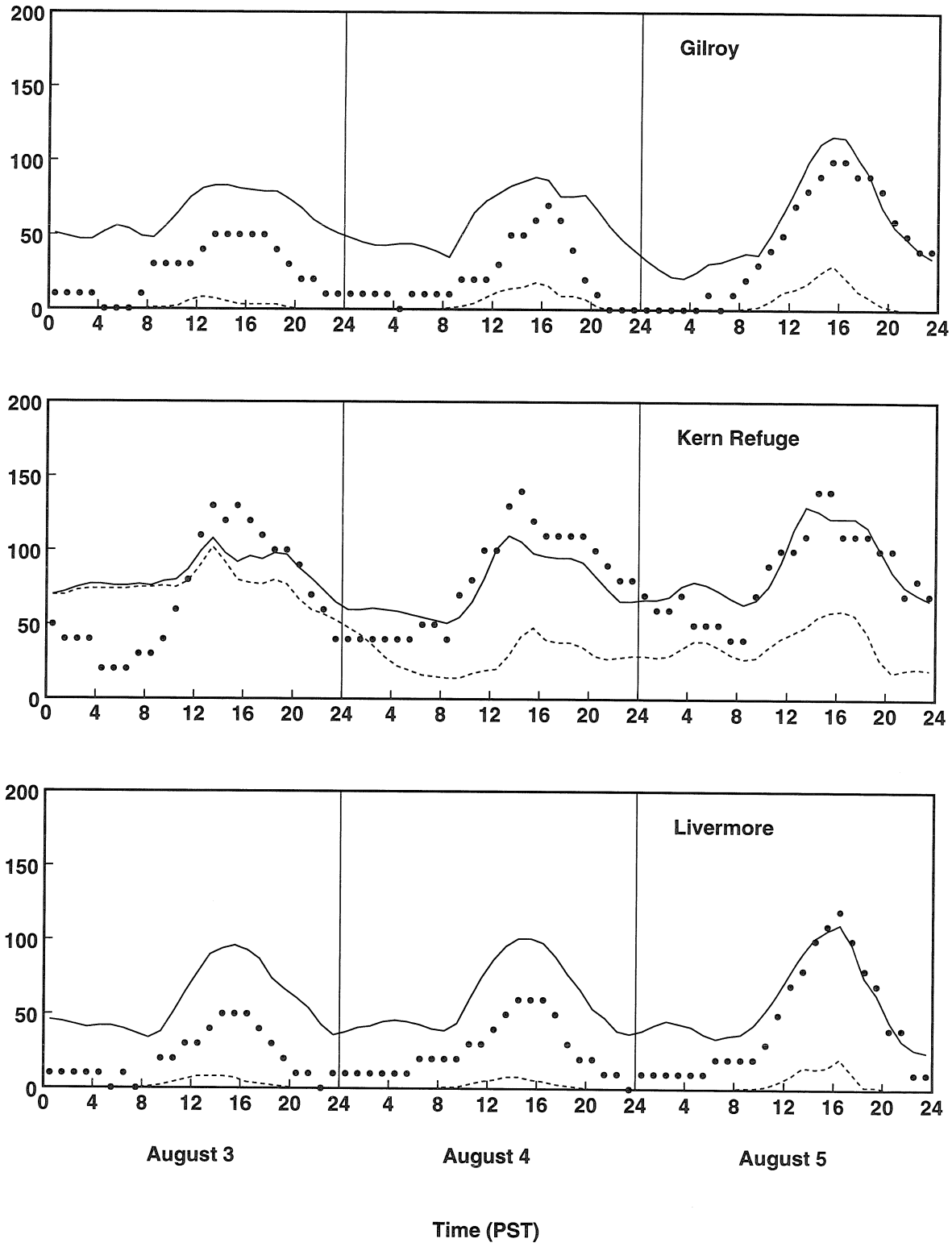


Figure 5.20: (Continued)

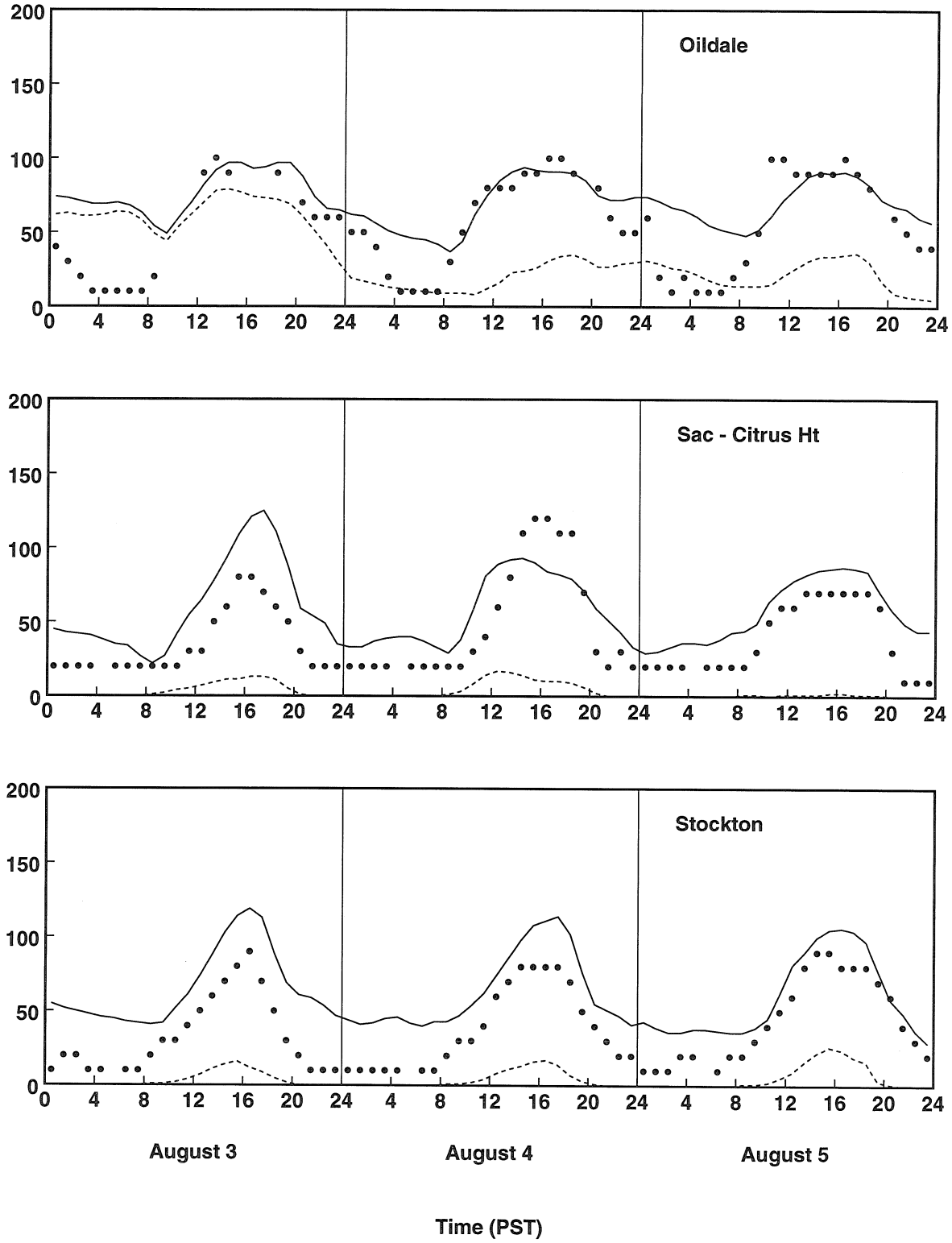


Figure 5.20: (Continued)

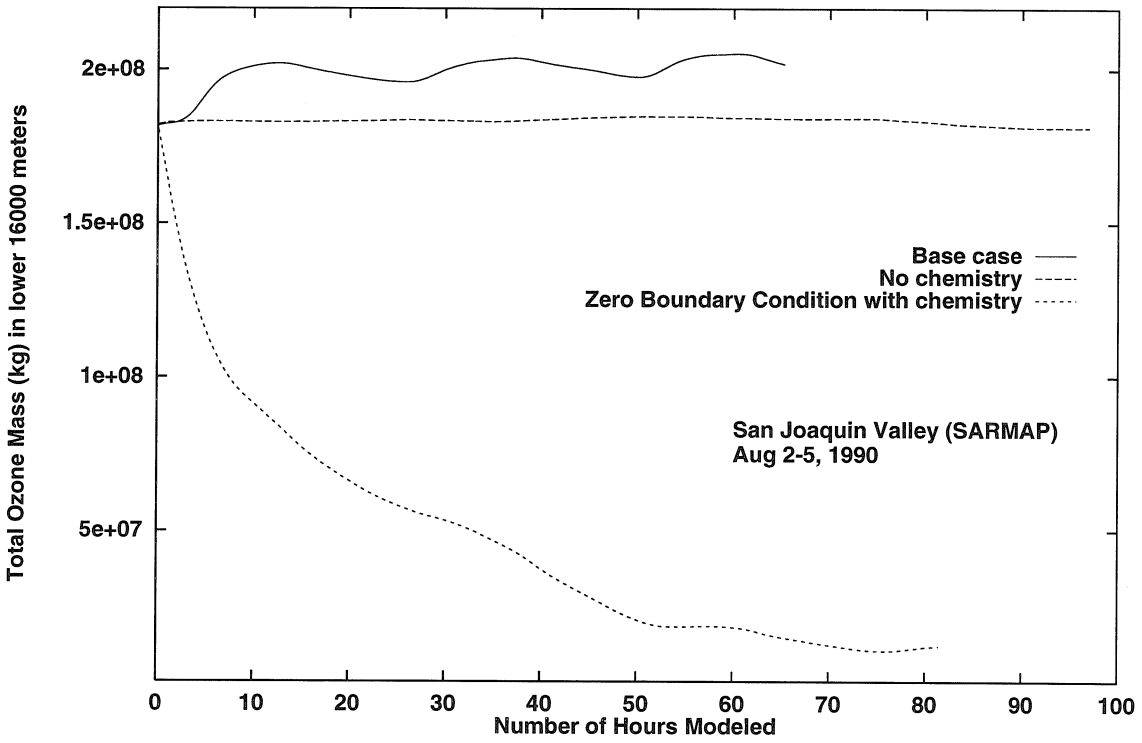


Figure 5.21: Total ozone mass in the San Joaquin Valley predicted by the SARMAP model. The solid line, large-dashed line, and short-dashed line correspond to the base case, no chemistry, and zero boundary condition runs respectively. The run starts at 0500 hrs of August 2, 1990.

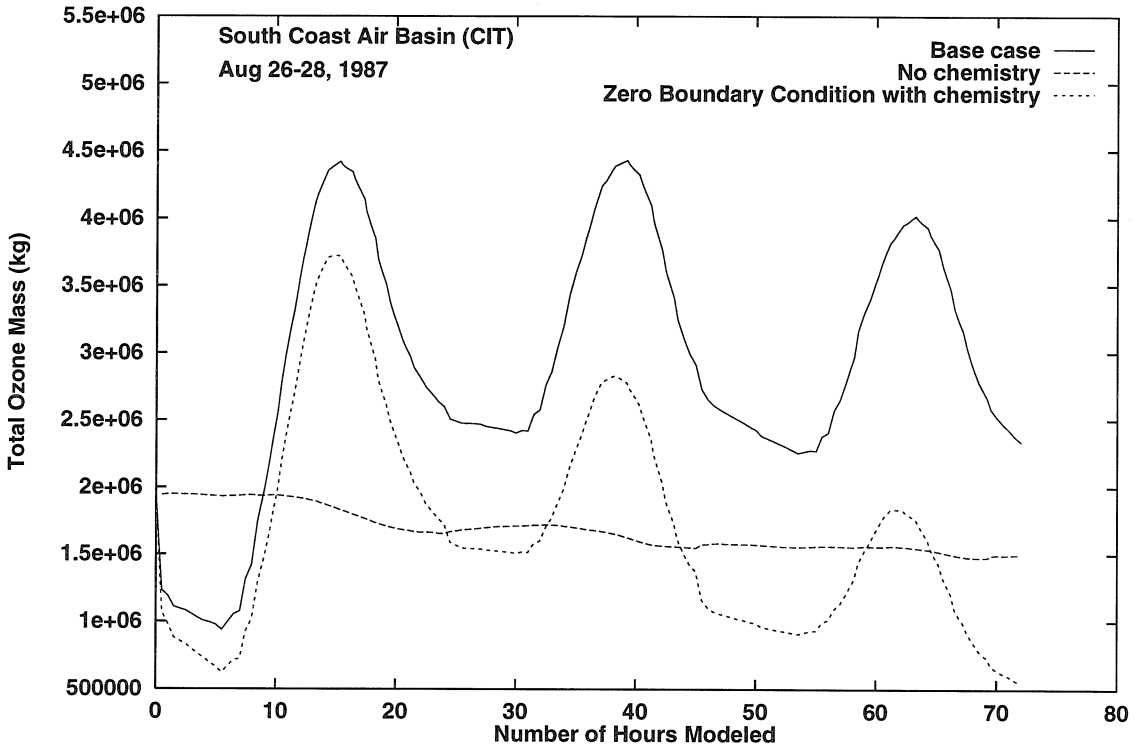


Figure 5.22: Total ozone mass in the South Coast Air Basin of California predicted by the CIT model. The solid line, large-dashed line, and short-dashed line correspond to the base case, no chemistry, and zero boundary condition runs respectively. The run starts at 0000 hrs of August 27, 1987.

the top of the modeling region used by SARMAP to model the San Joaquin Valley (~16,000 m) is significantly higher than that of the CIT code used to model the South Coast Air Basin (1,100 m). A higher vertical boundary in the model reduces the impact of ground emission while increasing the impact of boundary conditions on the model response. To examine this possibility, the total ozone mass present in the lower sections of the computational domain of the San Joaquin Valley is plotted in Figure 5.23. The total ozone mass predicted in the lower sections of the domain now presents peaks in the late afternoon hours that are indicative of a higher relative influence of emissions. However, Figure 5.23 shows that even for the lower 1,300 meters the influence of boundary conditions is still excessive. Namely, the short-dashed line still drops significantly relative to the base case run.

The high sensitivity of SARMAP might be due to the following reasons. First, the magnitude of the wind in the west boundary (the influx boundary of the domain) might be too high. Second, the level of the boundary conditions set might be too high. If either of these two hypotheses is true, that would explain the excessive decrease in total ozone mass when zeroing the boundary conditions. We will now proceed to study each case.

Figure 5.24 shows the average velocity of the incoming wind in the west boundary (Pacific Ocean) of both the San Joaquin Valley and the South Coast Air Basin. The wind speeds used in SARMAP to model the San Joaquin Valley are obtained prognostically using the meteorological model MM5 as described in previous sections. The wind speeds used in the CIT code to model the South Coast Air Basin of California are obtained diagnostically using interpolated values of actual wind measurements. The wind velocities of both models present the typical wind speed variations that arise from the diurnal temperature changes. The wind speeds produced by MM5 in the San Joaquin Valley are generally higher than those measured in the South Coast Air Basin at all times. The prognostic wind speeds generated by MM5 for the San Joaquin Valley must be evaluated relative to the actual observed wind speeds in the region.

Table 5.6 compares the maximum lateral boundary conditions used in the CIT

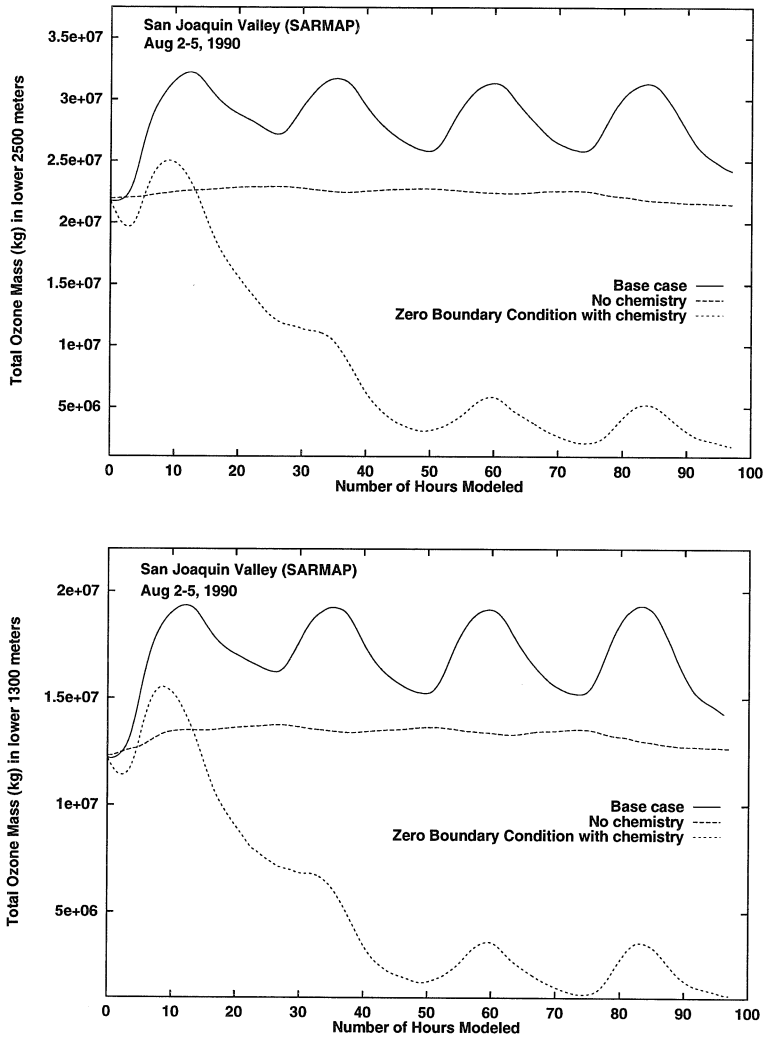


Figure 5.23: Total ozone mass in the lower regions of the San Joaquin Valley predicted by the SARMAP model. The solid line, large-dashed line, and short-dashed line correspond to the base case, no chemistry, and zero boundary condition runs respectively. The run starts at 0500 hrs of August 2, 1990.

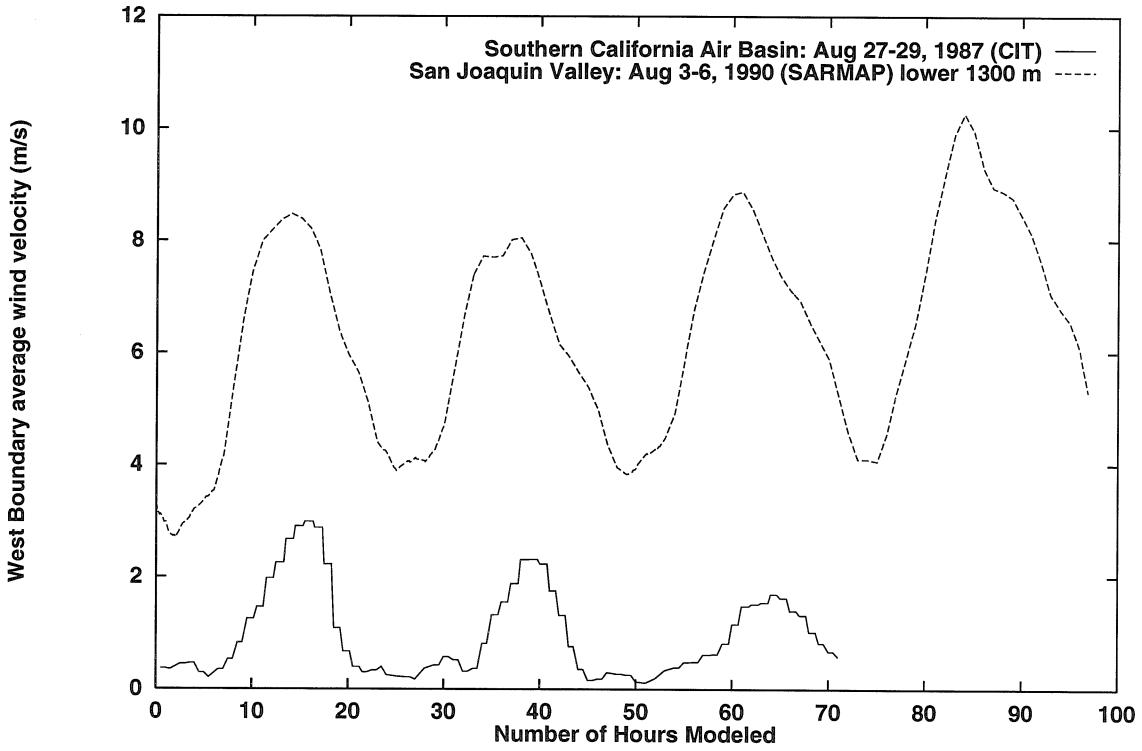


Figure 5.24: Average wind velocity in the lower 1,300 m of the San Joaquin Valley and the South Coast Air Basin of California.

model and the SARMAP model. The comparison is not one-to-one since the two models use different chemical mechanisms. The boundary conditions of most chemically similar species are of comparable magnitude in the two models. Nevertheless, the boundary conditions of SO2 and PAR used in SARMAP stand out as being too high. To examine further this possibility, the ratio of total mass flux across the boundaries to total emissions in the modeling region is computed between the two models and shown in Tables 5.7, 5.8 and 5.9. The influx mass to emitted mass ratio computed in Table 5.8 is smaller than the same ratio in Table 5.7 because the total mass emitted remains constant (emissions are near ground levels) while the mass flux from boundary conditions decreases as the top of modeling region decreases. Comparing the ratios of SARMAP given in Table 5.8 with those of the CIT model given in Table 5.9, SO2 and HCHO present the greatest discrepancies. As it is shown in Table 5.6 the boundary condition of HCHO is about two times higher in SARMAP than in the CIT model. The ratio of HCHO of SARMAP is large since the total mass of HCHO emitted is small. On the other hand, the ratio of SO2 in SARMAP is significantly greater than in the CIT model due to the large mass of SO2 that enters the domain from the boundary conditions. The boundary condition of SO2 and PAR used in SARMAP must be compared with actual observed concentrations.

The previous analysis presents the following puzzling question: If some of the boundary conditions used in SARMAP are overestimated and if the entire advective field is also too high, why do the ozone levels predicted by the model base case run match so well with observed data? To address that question one must examine separately the effect of each phenomenon on the overall dynamics of the region.

First, the effect of decreasing the wind speed will be discussed. Intuitively, by decreasing the wind speed one would expect an increase in ozone concentrations. Reducing the wind speed is equivalent to increasing the residence time of the "atmospheric reactor." A lower wind velocity would provide a longer time for emissions to be present inside the modeling domain. Figure 5.25 shows the ozone levels predicted by the base case run and those predicted by reducing the wind by a factor of 4 using the GALK solver. As expected, the ozone predicted by the reduced wind run in all

Table 5.6: Maximum boundary condition comparison between the CIT and SARMAP models.

Species CIT[a]	B.C. (ppb)	Species SARMAP[b]	B.C. (ppb)
CO	200.0	CO	200.0
NO	1.0	NO	0.5
NO2	1.0	NO2	2.5
ETHE	1.7	ETH	1.4
HCHO	3.0	HCHO	5.8
ISOP	0.0	ISO	0.1
SO2	1.0	SO2	10.0
SO3	0.0	SULF	0.1
TOLU	1.5	TOLU	0.5
ALD2	0.5	ALD	1.5
ALKE	1.8	OLE	0.8
ALKA	9.5	PAR	42.0
AROM	1.6	XYL	0.3
MEK	4.0		

[a] LCC chemical mechanism.
[b] CBM-IV chemical mechanism.

Table 5.7: Total emissions and total mass flow in all layers of the SARMAP model (~16,000 m).

Species	Emiss (kg)[a]	B.C. (kg)[b]	ratio
SO2	646355	68114910	105.3
SULF	25321	1021723	40.3
NO2	511911	12188985	23.8
NO	2990076	1591626	0.5
ALD	1390981	7148363	5.1
HCHO	85979	18453062	214.6
PAR	7957446	62721006	7.8
ETH	435830	4182247	9.5
OLE	800310	2466738	3.0
ISO	2408838	723720	0.3
TOL	439984	4802389	10.9
XYL	505713	3053858	6.0
CO	27319818	2536474013	92.8
HONO	25605	50021	1.9

[a] August 3-6, 1990 total emissions in modeling region.
[b] August 3-6, 1990 total mass flowing in across boundaries.

Table 5.8: Total emissions and total mass flow in the first 8 layers of the SARMAP model (~2,500 m).

Species	Emiss (kg)	B.C. (kg)	ratio
SO2	646355	40023740	61.9
SULF	25321	600356	23.7
NO2	511911	7191765	14.0
NO	2990076	938056	0.3
ALD	1390981	4193487	3.0
HCHO	85979	10851436	126.2
PAR	7957446	36806330	4.6
ETH	435830	2458458	5.6
OLE	800310	1449859	1.8
ISO	2408838	425252	0.1
TOL	439984	2830679	6.4
XYL	505713	1789811	3.5
CO	27319818	350207725	12.8
HONO	25605	29392	1.1

[a] August 3-6, 1990 total emissions in modeling region.
[b] August 3-6, 1990 total mass flowing in across boundaries.

stations tends to be higher than the base case run. Furthermore, one would intuitively expect that by reducing the wind speed the effect of zeroing boundary conditions (as shown in Figure 5.20) would also be reduced. Figure 5.26 shows the ozone levels predicted by reducing the wind by a factor of 4 using the GALK solver with those of the same run with zero boundary conditions. When zeroing the boundary conditions in Figure 5.26 the ozone levels are slightly smaller than with non zero boundary conditions. However, the ozone reduction obtained by decreasing the boundary conditions is not as excessive as the one shown in Figure 5.20, which reduces the ozone levels almost entirely in some locations such as Livermore and Stockton.

The data used to generate Figure 5.25 and Figure 5.26 was obtained using GALK as the horizontal advection solver. The BOTT schemes becomes unstable during the third day of the simulation. Decreasing the wind speeds decreases the Courant number of the problem. The BOTT solver becomes unstable at lower Courant numbers. The data generated by performing the same run with the BOTT solver also confirms (before it becomes unstable) that the ozone levels increase by reducing the wind

Table 5.9: Total emissions and total mass flow in all layers of the CIT model (1,100 m).

Species	Emiss (kg)	B.C. (kg)	ratio
CO	5575004	6255394	1.12
NO	696032	33916	0.05
NO2	56168	51826	0.92
ETHE	138261	53892	0.39
ALKE	150018	95136	0.63
ALKA	943418	873244	0.92
TOLU	216103	156235	0.72
AROM	176220	201425	1.14
HCHO	32790	101170	3.09
ALD2	20779	257342	12.50
MEK	50586	325620	6.45
MEOH	4734	0	0
ETOH	99633	0	0
ISOP	82545	0	0
NH3	166414	0	0
SO2	108037	201425	1.87
SO3	3918	0	0

[a] August 27, 1987 total emissions in modeling region.
[b] August 27, 1987 total mass flowing in across boundaries.

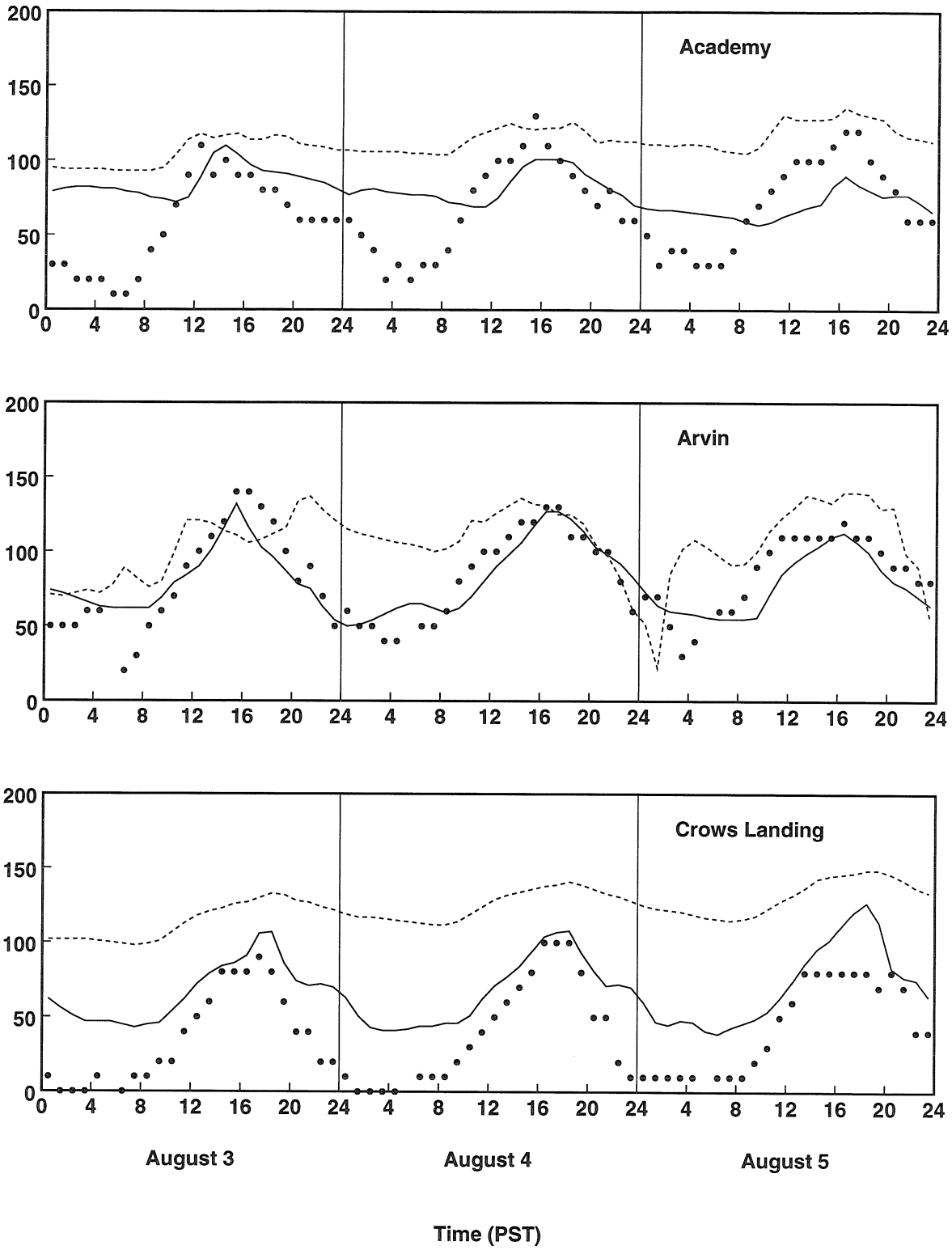


Figure 5.25: Time series plot of observed ozone concentrations in ppb (solid circles), base case ozone predictions (solid line) and base case with wind reduced by a factor of 4 and no O3 deposition (dashed line).

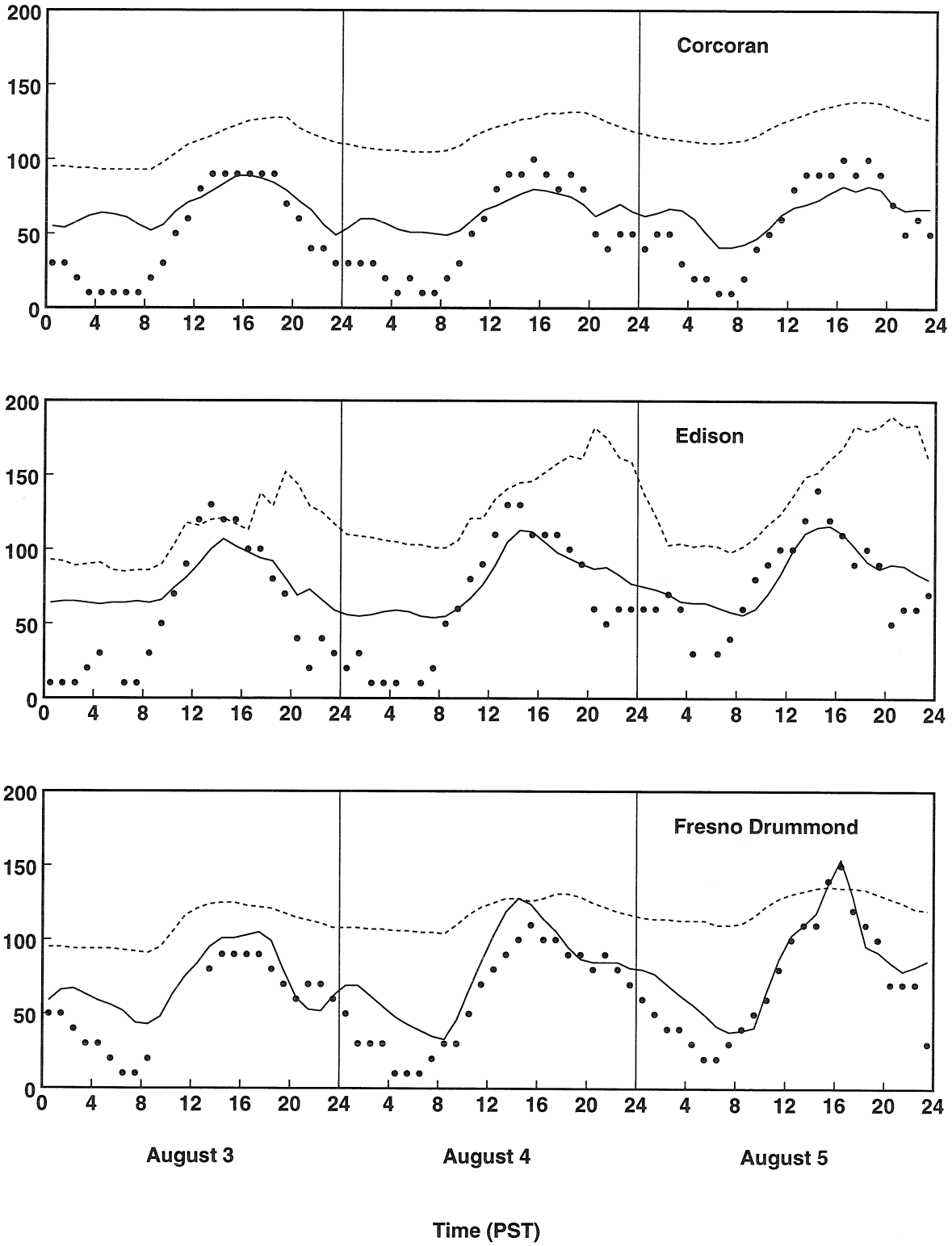


Figure 5.25: (Continued)

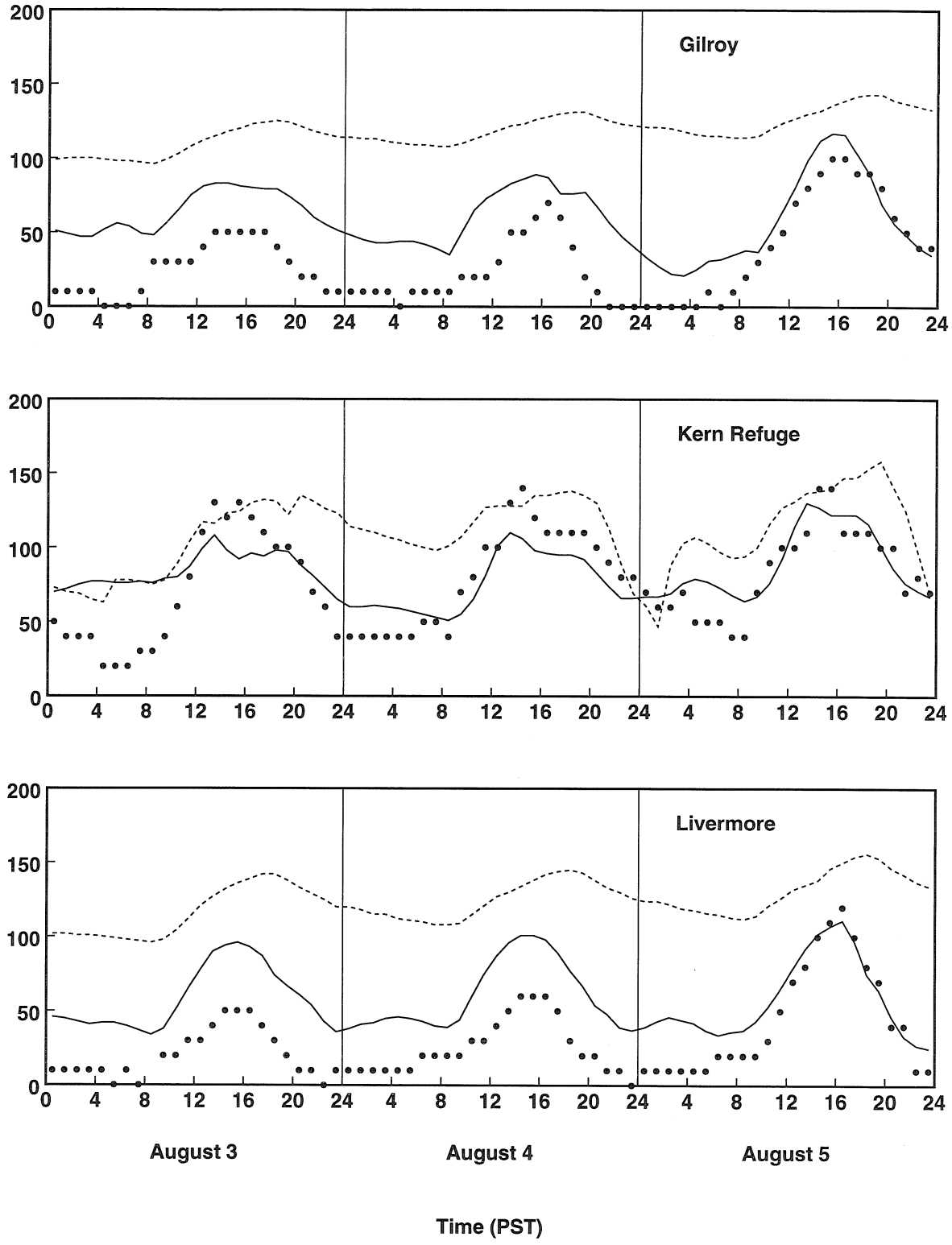


Figure 5.25: (Continued)

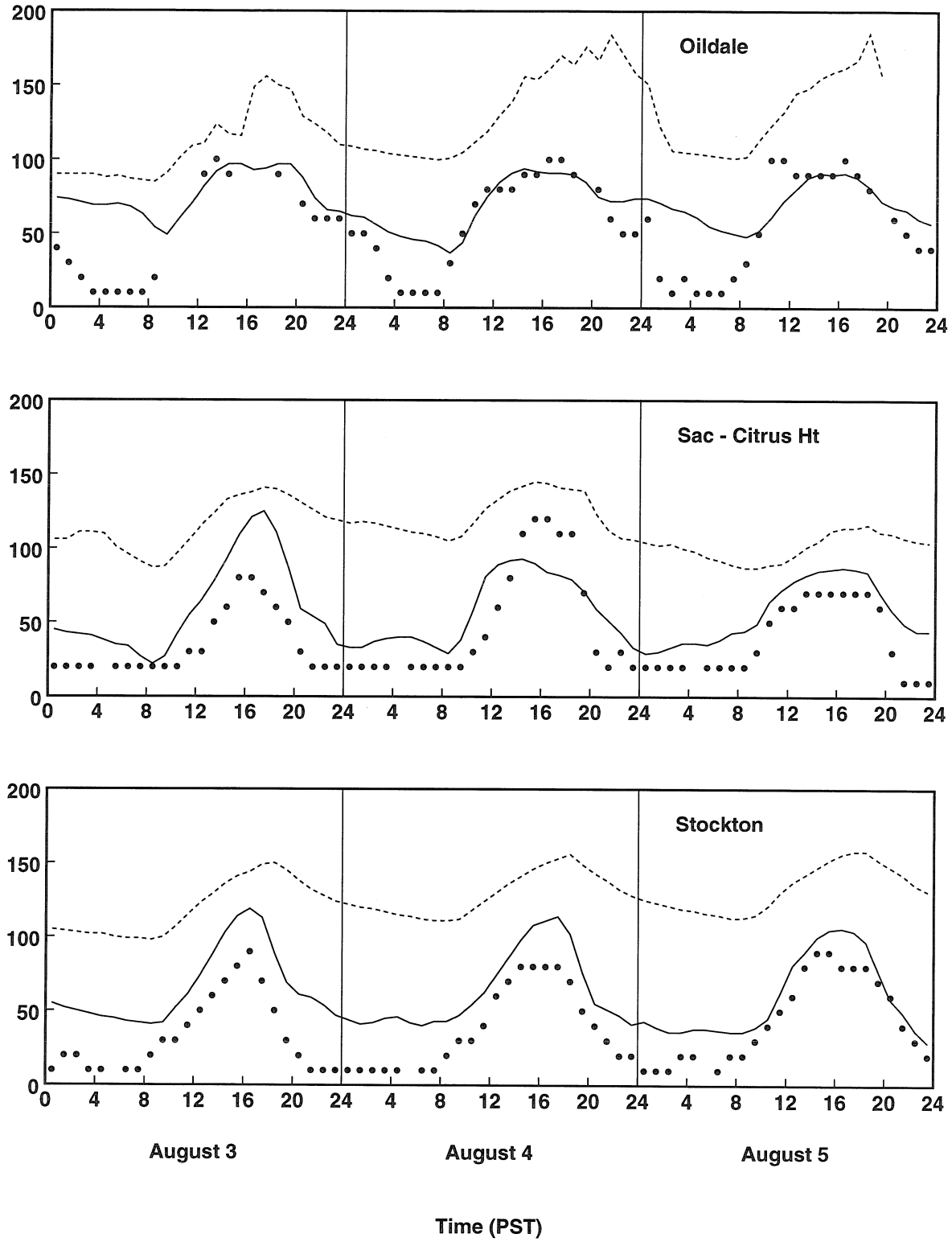


Figure 5.25: (Continued)

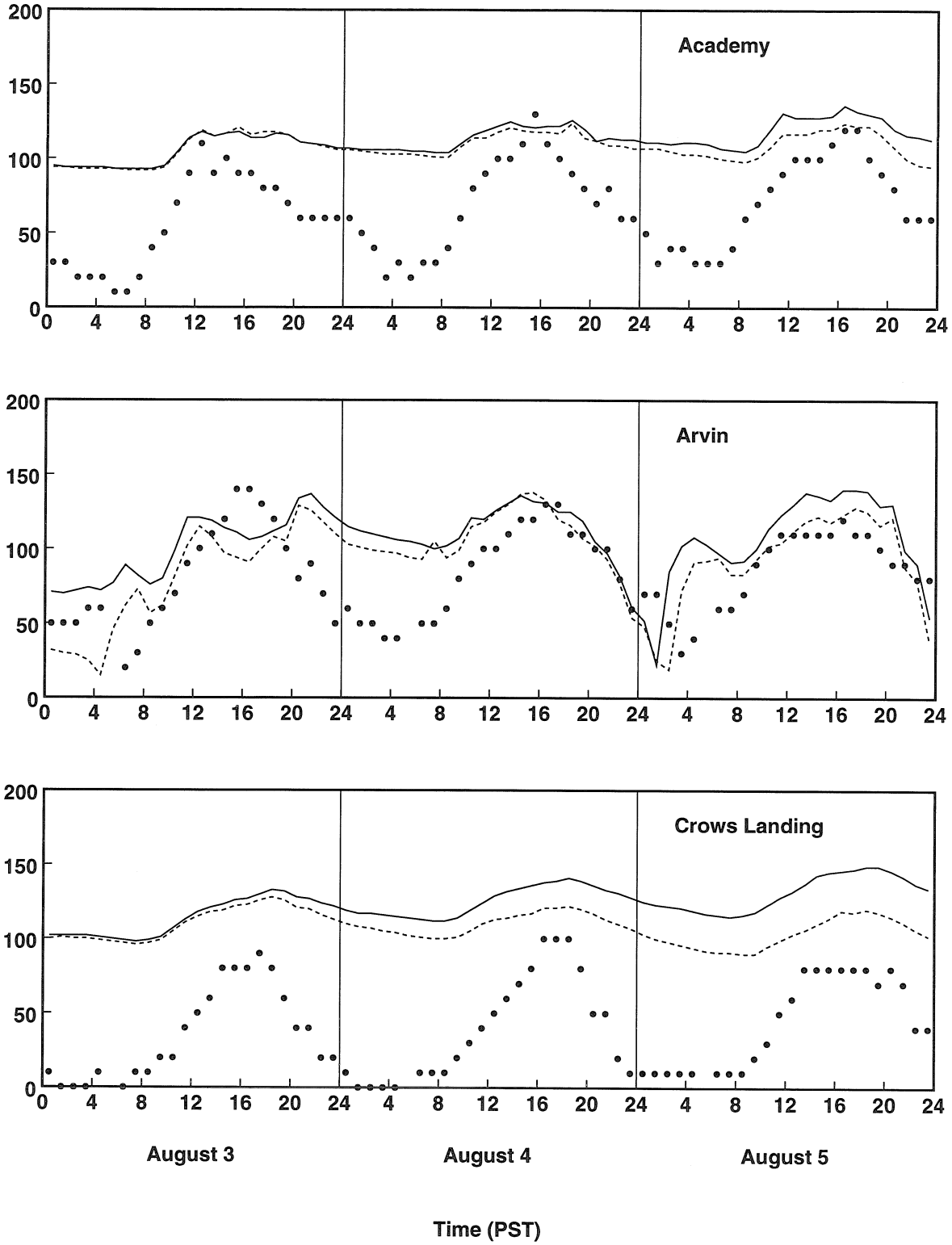


Figure 5.26: Time series plot of observed ozone concentrations in ppb (solid circles), base case with wind reduced by a factor of 4 and no O3 deposition (solid line) and base case with wind reduced by a factor of 4, no O3 deposition and zero boundary conditions (dashed line).

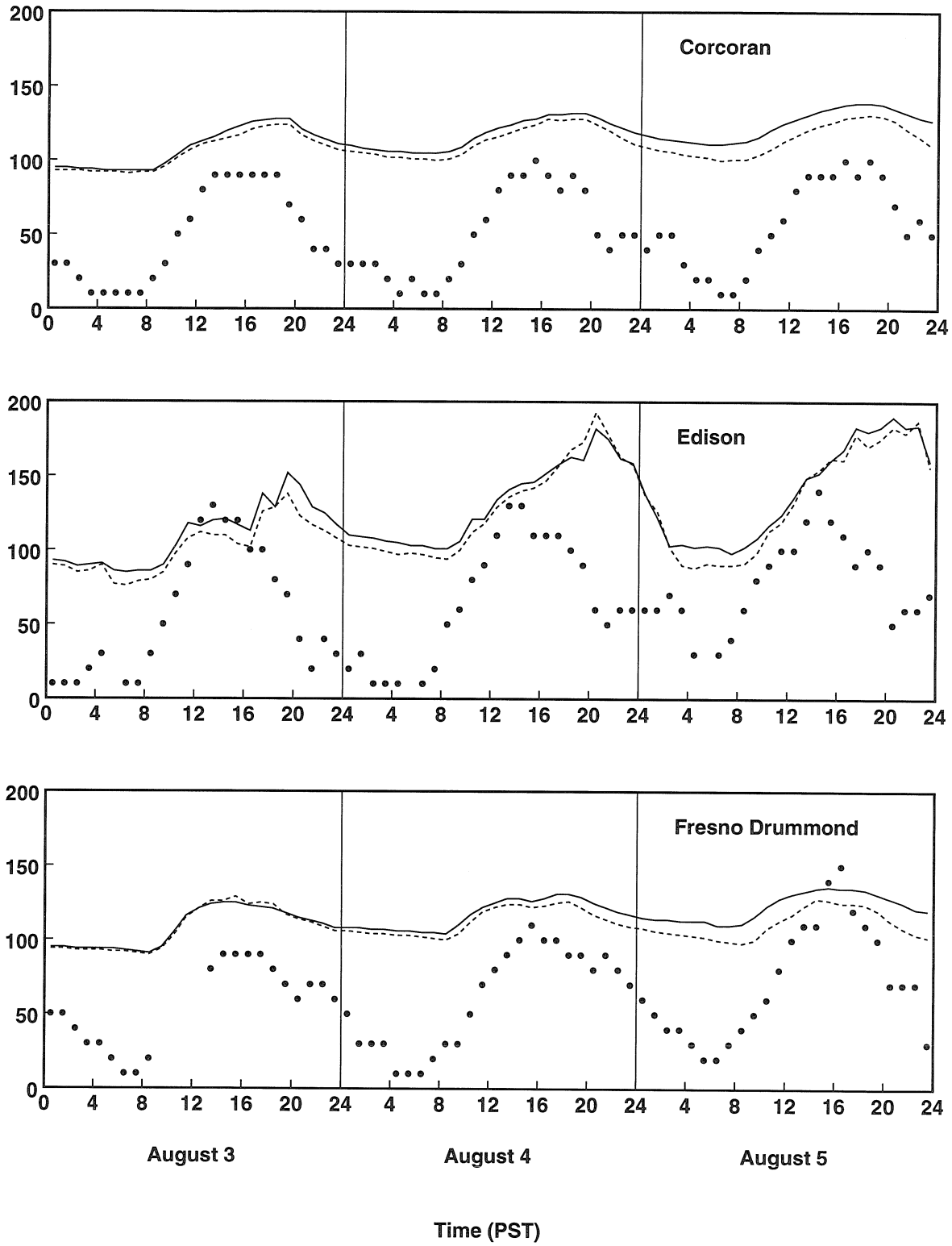


Figure 5.26: (Continued)

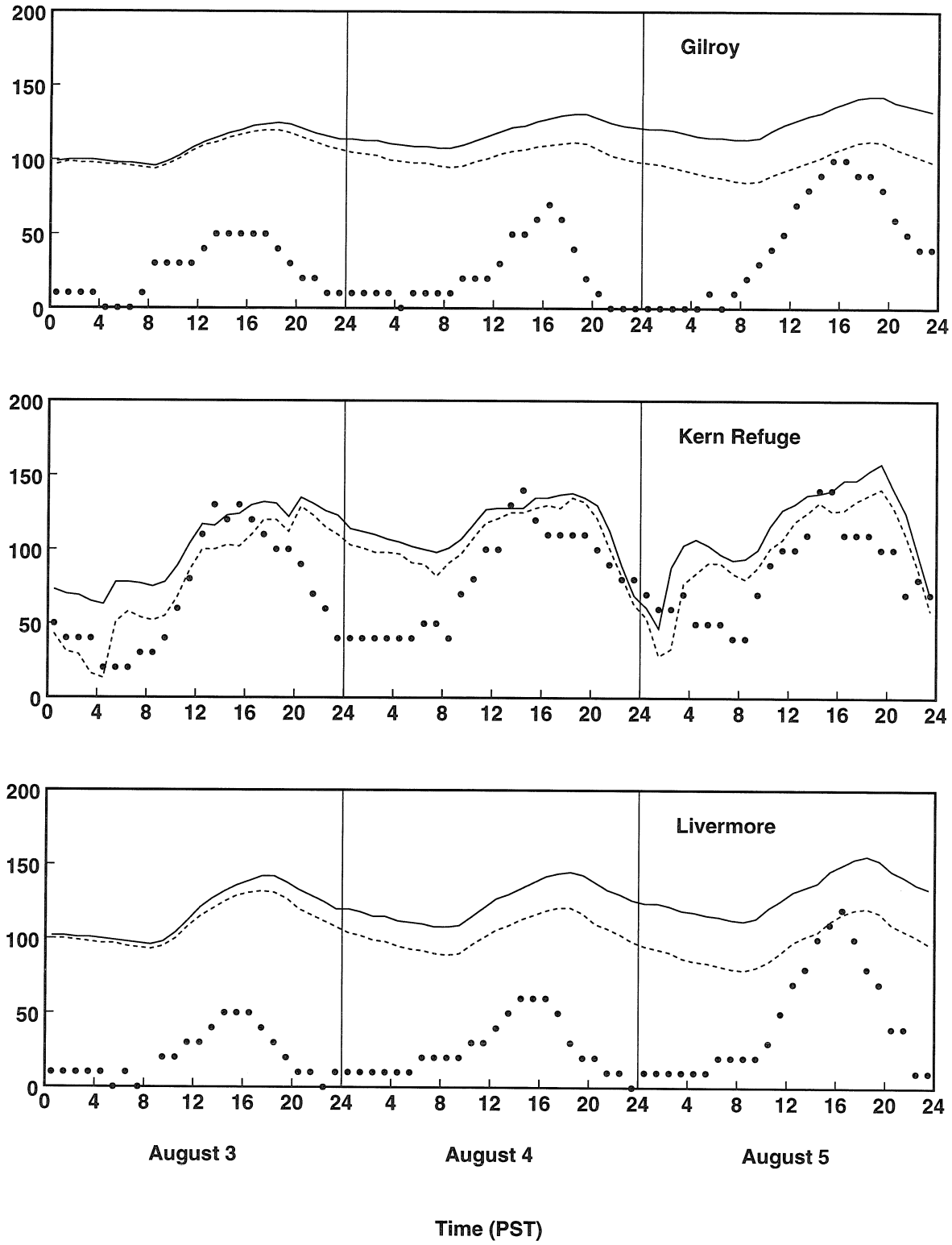


Figure 5.26: (Continued)

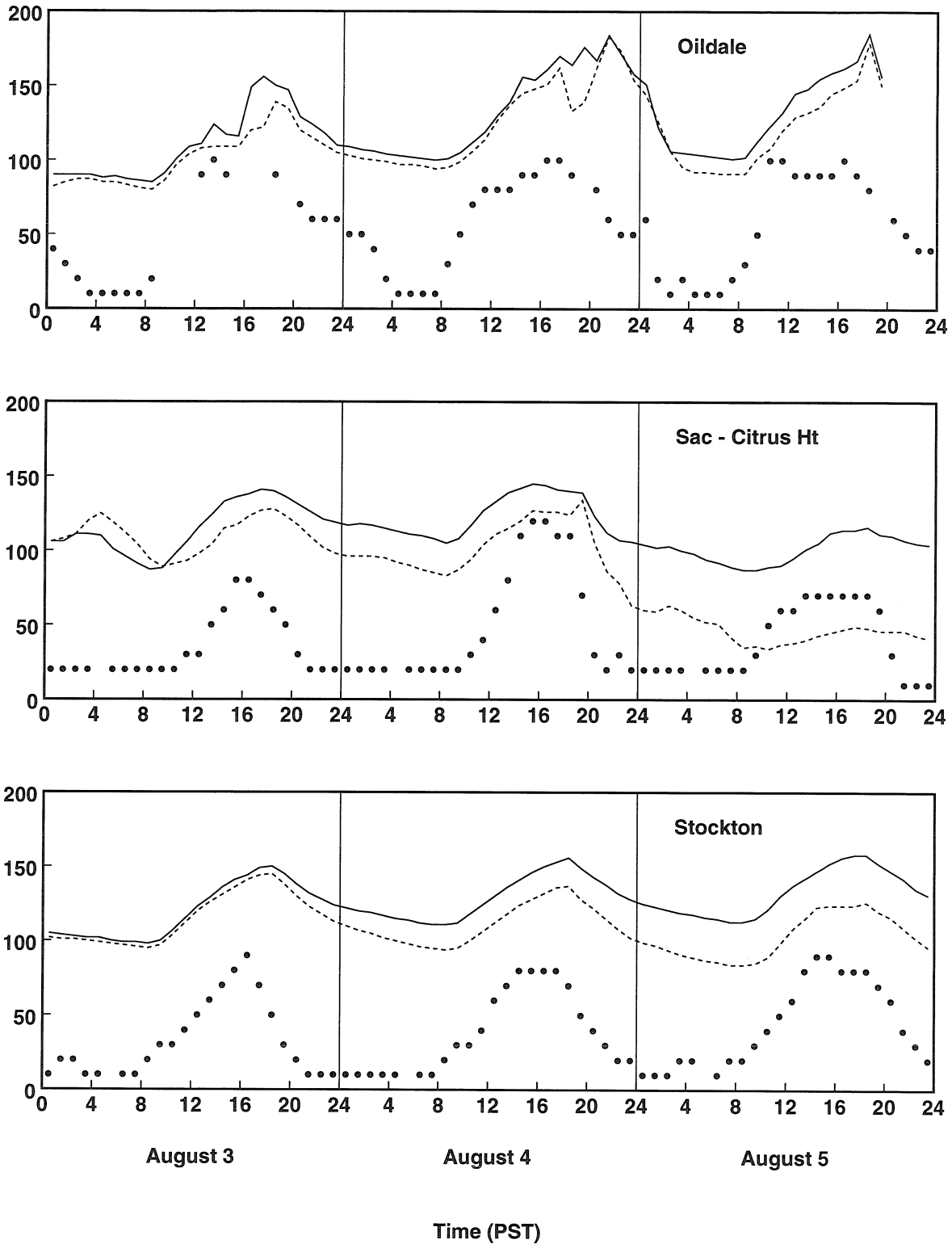


Figure 5.26: (Continued)

speed. The deposition of O3 was zeroed to generate Figures 5.25 and 5.26 to focus only on the effect of wind reduction. Figure 5.27 compares the base case ozone predictions with a reduced wind (factor of 4) run with O3 deposition. The increase of ozone predicted is still observed in stations located near influx boundaries such as Citrus Heights, Stockton, Livermore, Crows Landing and Gilroy.

Figure 5.28 shows the base case ozone predictions and the base case with wind reduced by a factor of 2, with O3 deposition. Ozone levels in the downwind stations are very similar to the run when the wind is reduced by a factor of 4. Ozone levels in the upwind stations are slightly lower than the run when the wind is reduced by a factor of 2 but higher than the base case run. The trend of ozone reduction by lowering the residence time of the "atmospheric reactor" (increasing the wind speed) is confirmed.

It has been shown that by reducing the wind speed only, the ozone levels tend to be overpredicted in the upwind locations. However, there is one more issue that needs to be addressed: the boundary values of SO2 and PAR concentrations used in SARMAP. Figure 5.29 shows base case ozone predictions and the base case with wind reduced by a factor of 2, with O3 deposition, the SO2 and PAR boundary values reduced by a factor of 10 and 5 respectively. It is observed that the agreement between observations and the ozone predictions using the reduced wind and reduced boundary conditions are comparable to the good agreement that was obtained previously with high winds and high boundary conditions. Furthermore, Figure 5.30 shows that the excessive sensitivity of ozone to zero boundary conditions is significantly reduced once the wind and boundary data is modified. The solid line shows the typical peaks in the afternoon hours. The large-dashed line, which corresponds to the base case run with no chemistry, slightly decreases as the O3 is being deposited to the ground. The short-dashed line, which correspond to zero boundary conditions with chemistry, decreases with time as new clean air enters in the domain. Figure 5.30 is to be compared with Figure 5.23 which presents an abrupt decrease in the total ozone of the domain once the boundary conditions have been zeroed.

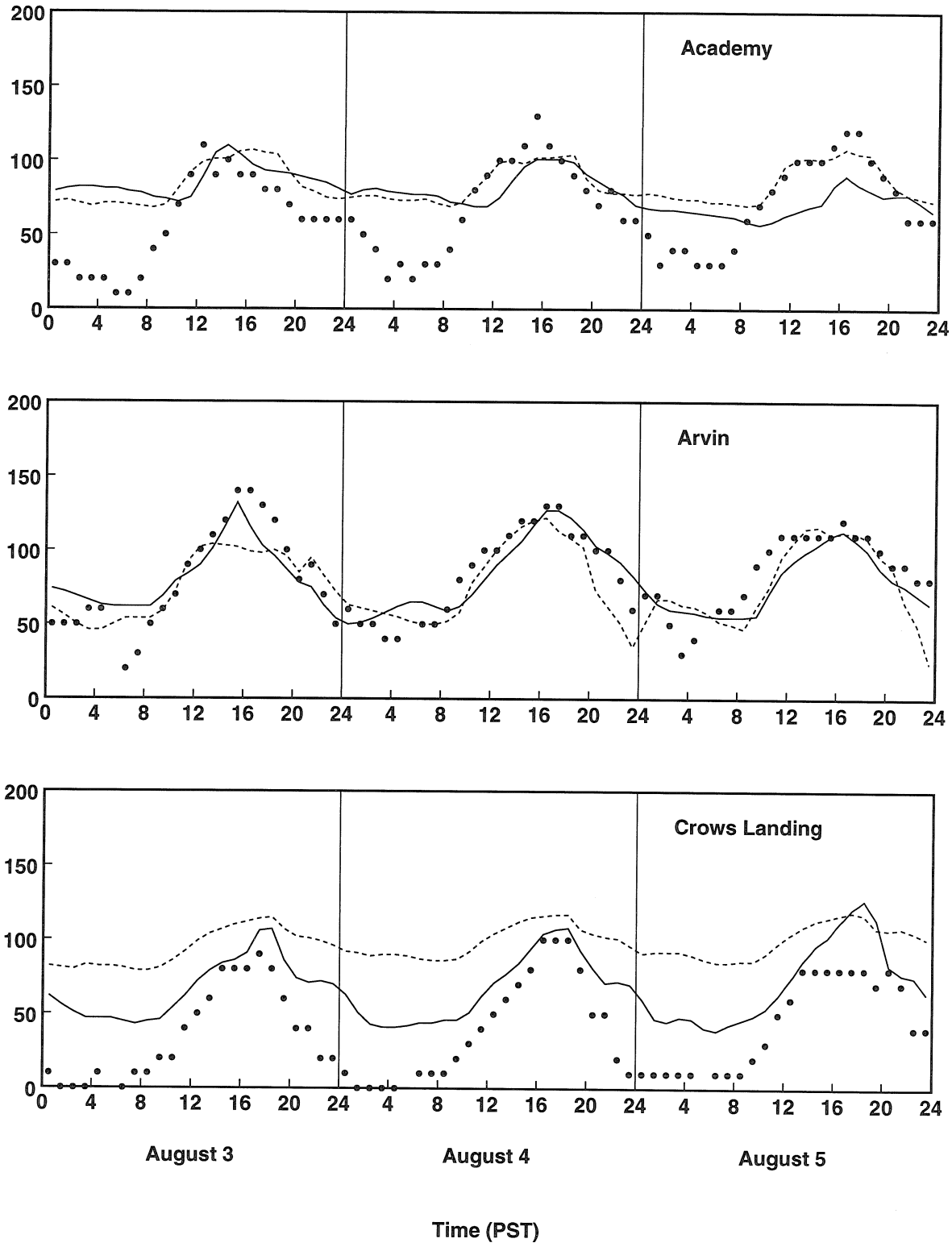


Figure 5.27: Time series plot of observed ozone concentrations in ppb (solid circles), base case ozone predictions (solid line) and base case with wind reduced by a factor of 4, with O3 deposition (dashed line).

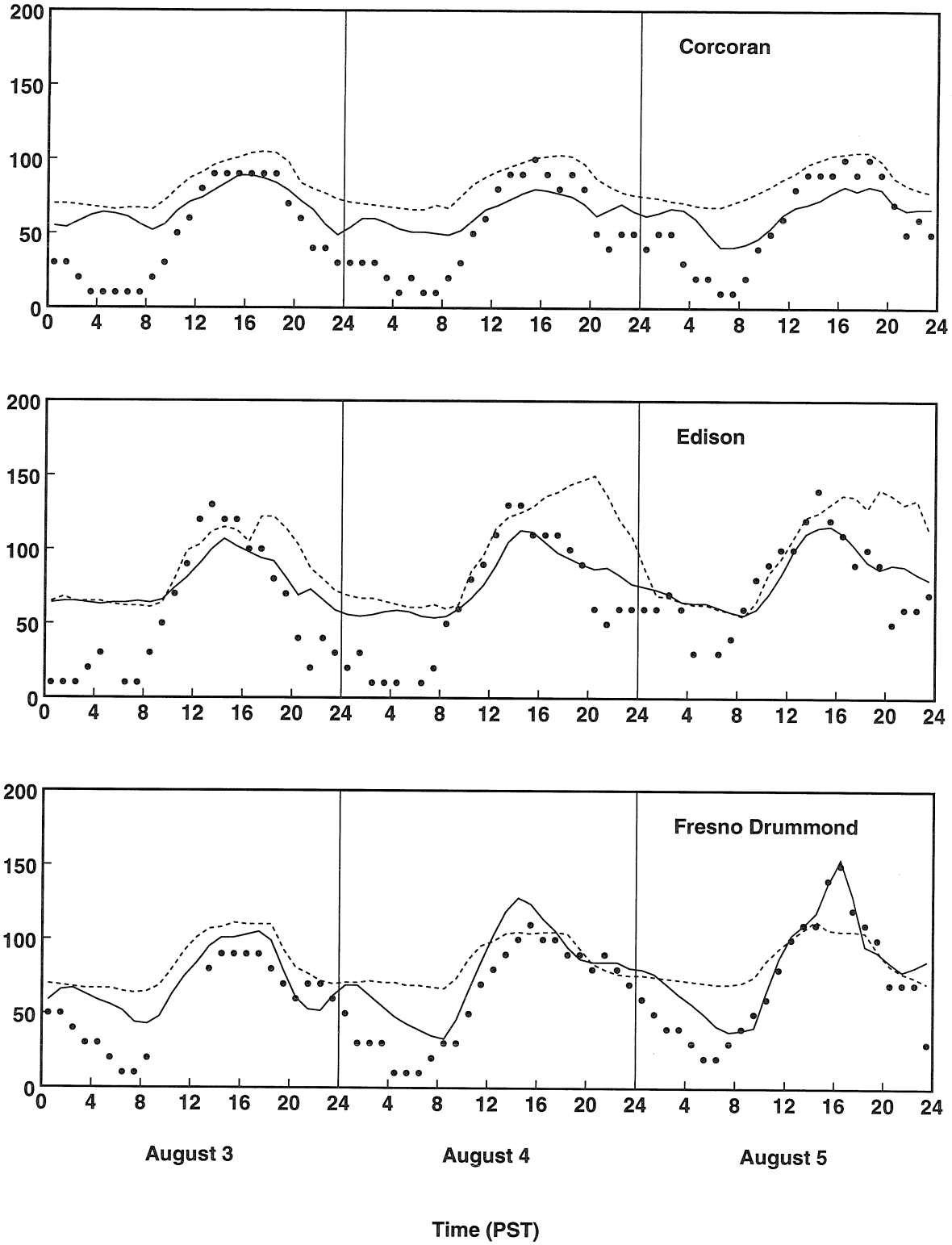


Figure 5.27: (Continued)

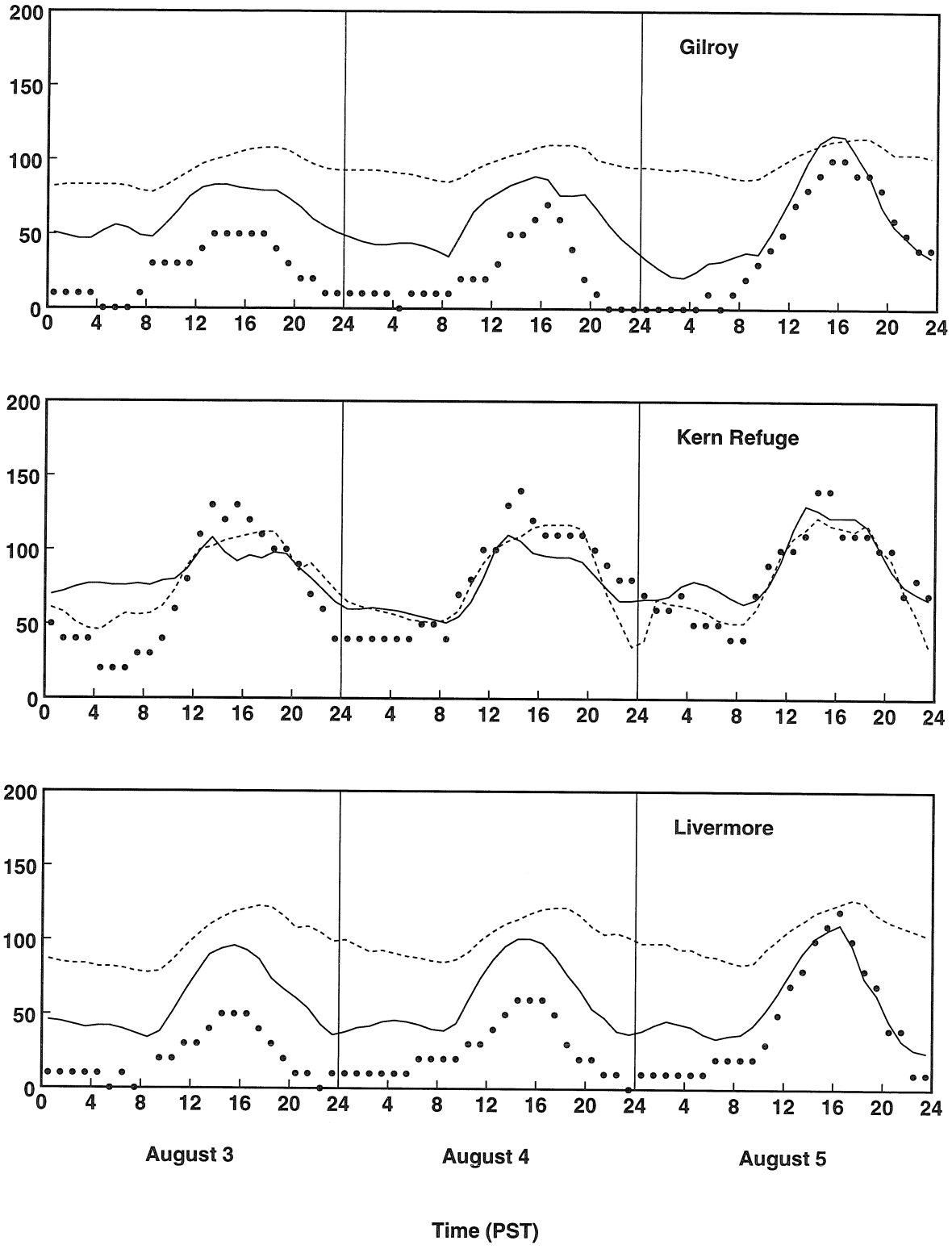


Figure 5.27: (Continued)

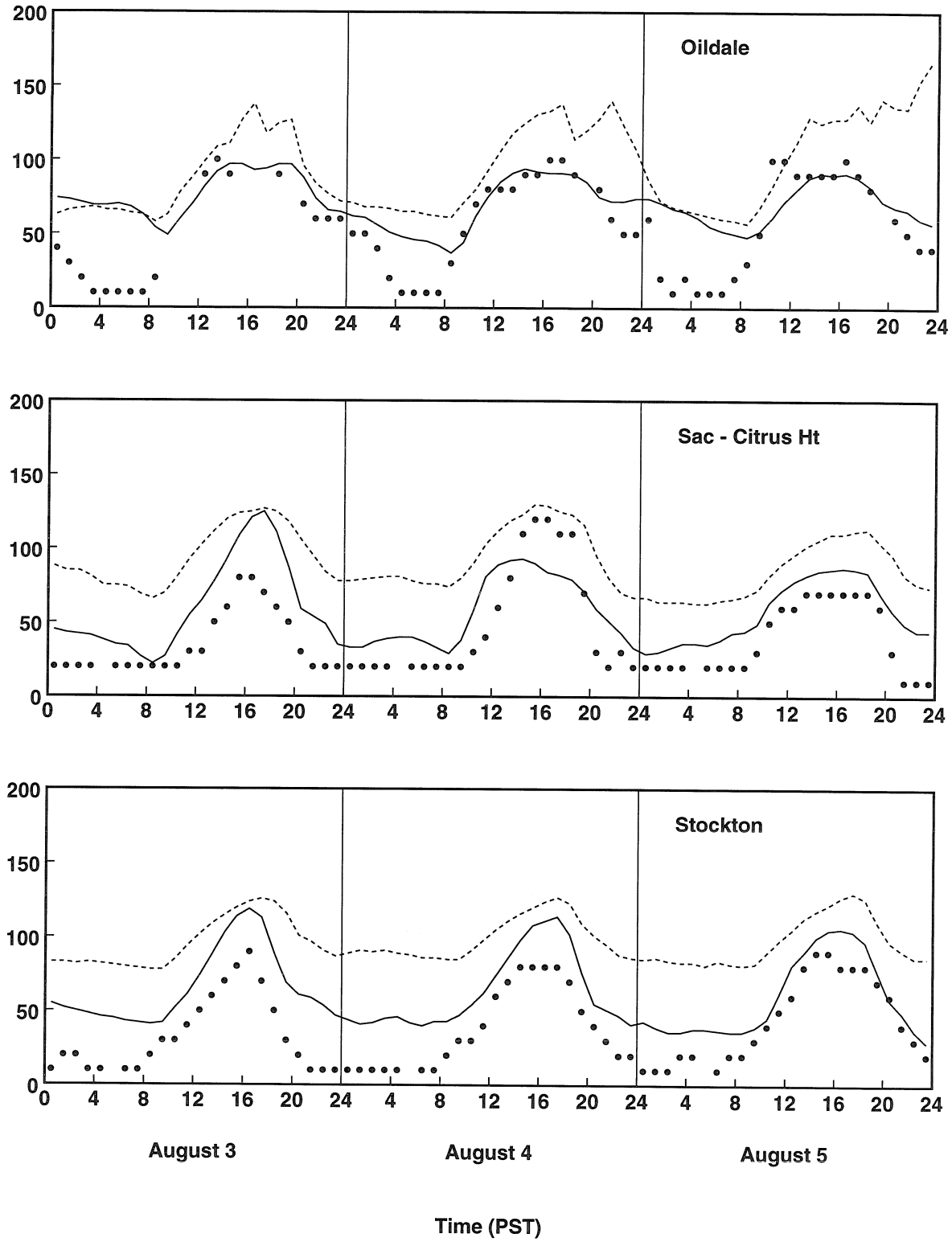


Figure 5.27: (Continued)

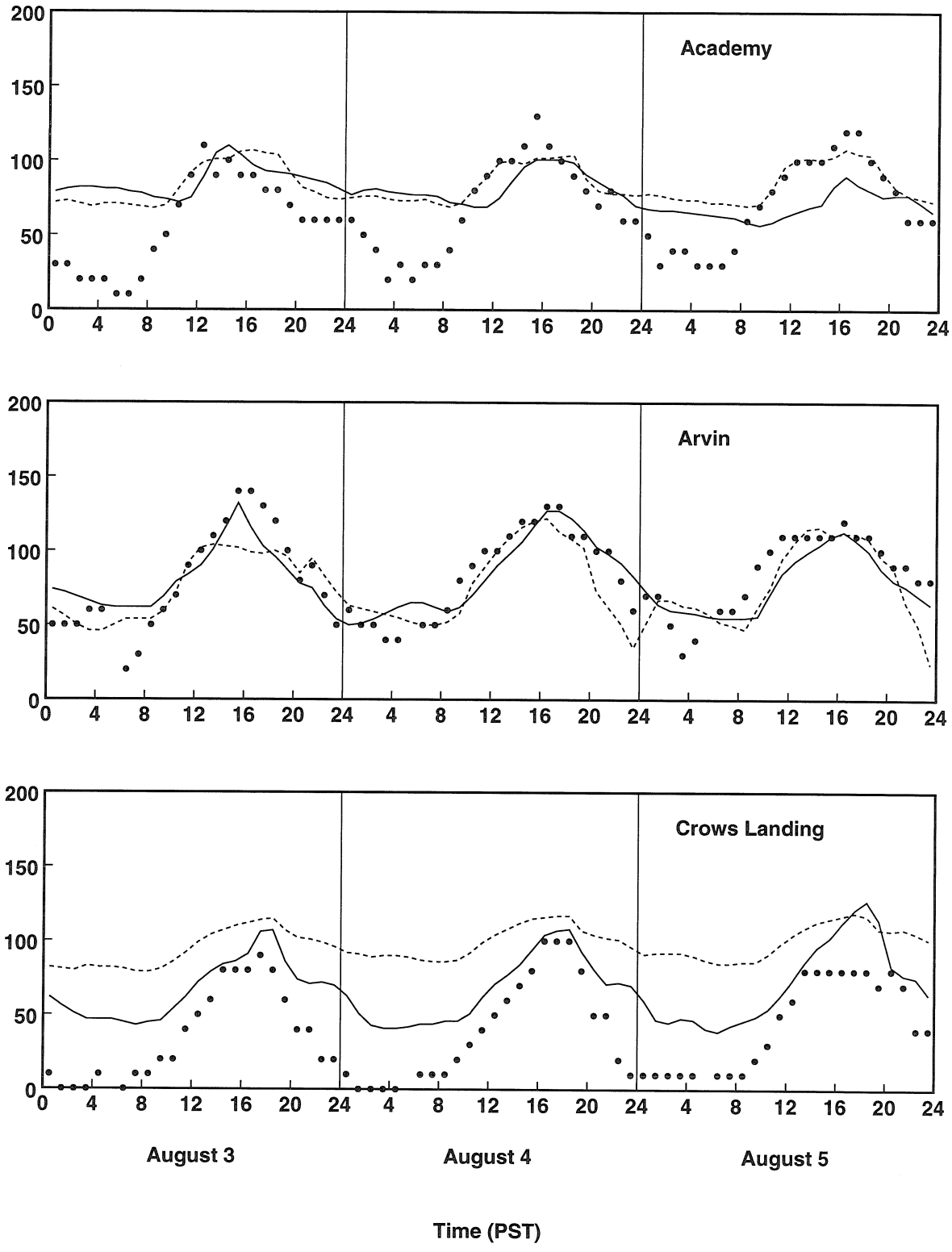


Figure 5.28: Time series plot of observed ozone concentrations in ppb (solid circles), base case ozone predictions (solid line) and base case with wind reduced by a factor of 2, with O3 deposition (dashed line).

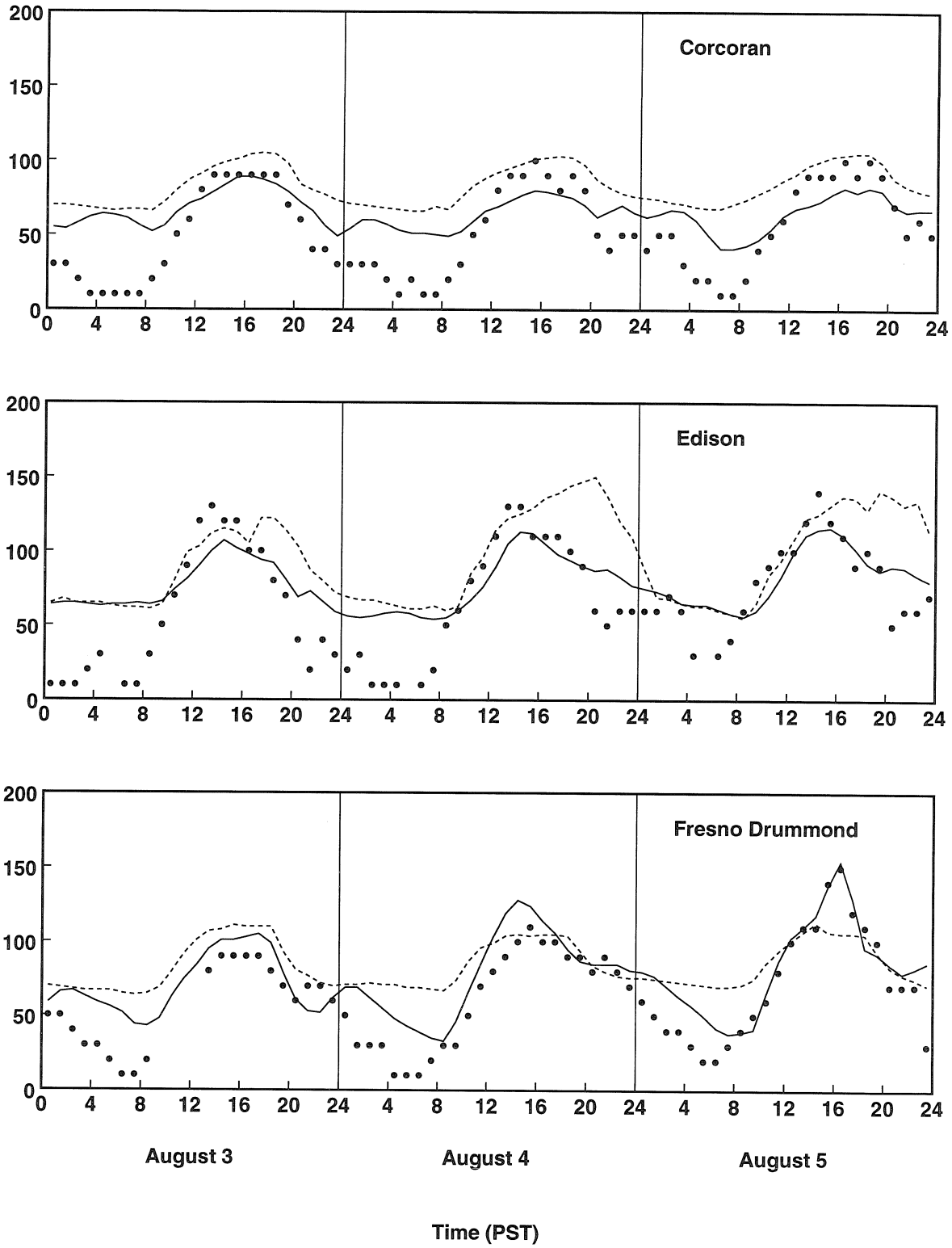


Figure 5.28: (Continued)

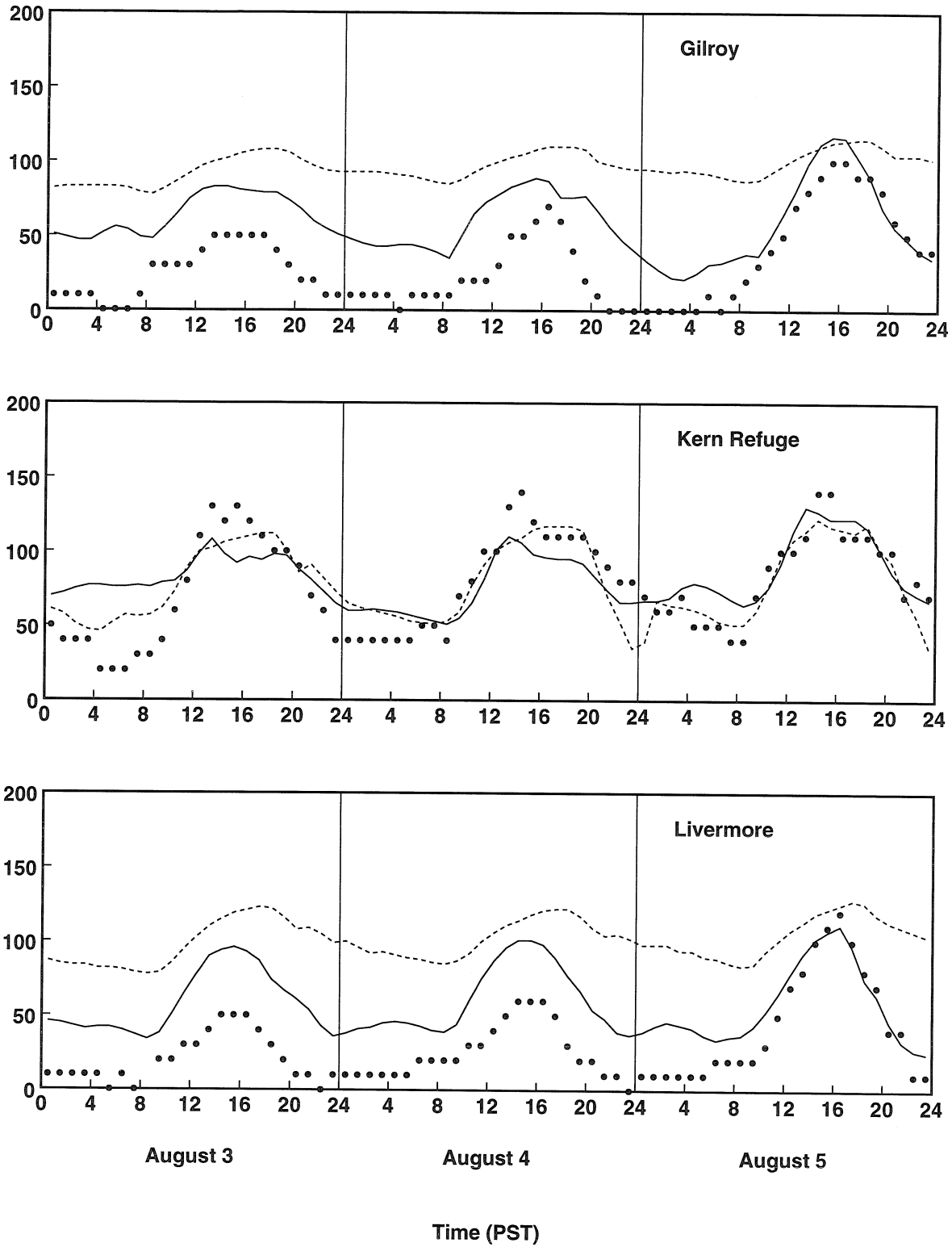


Figure 5.28: (Continued)

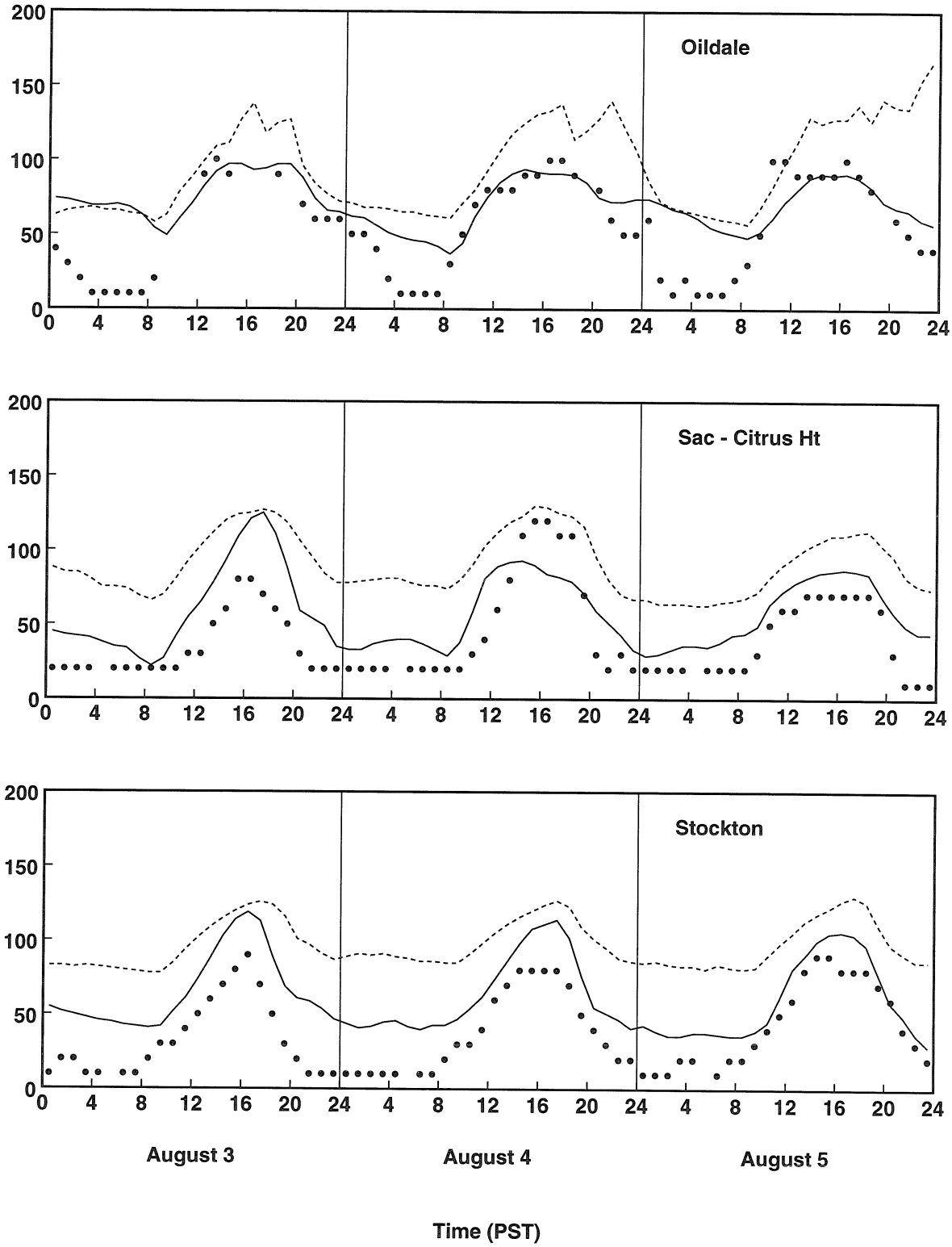


Figure 5.28: (Continued)

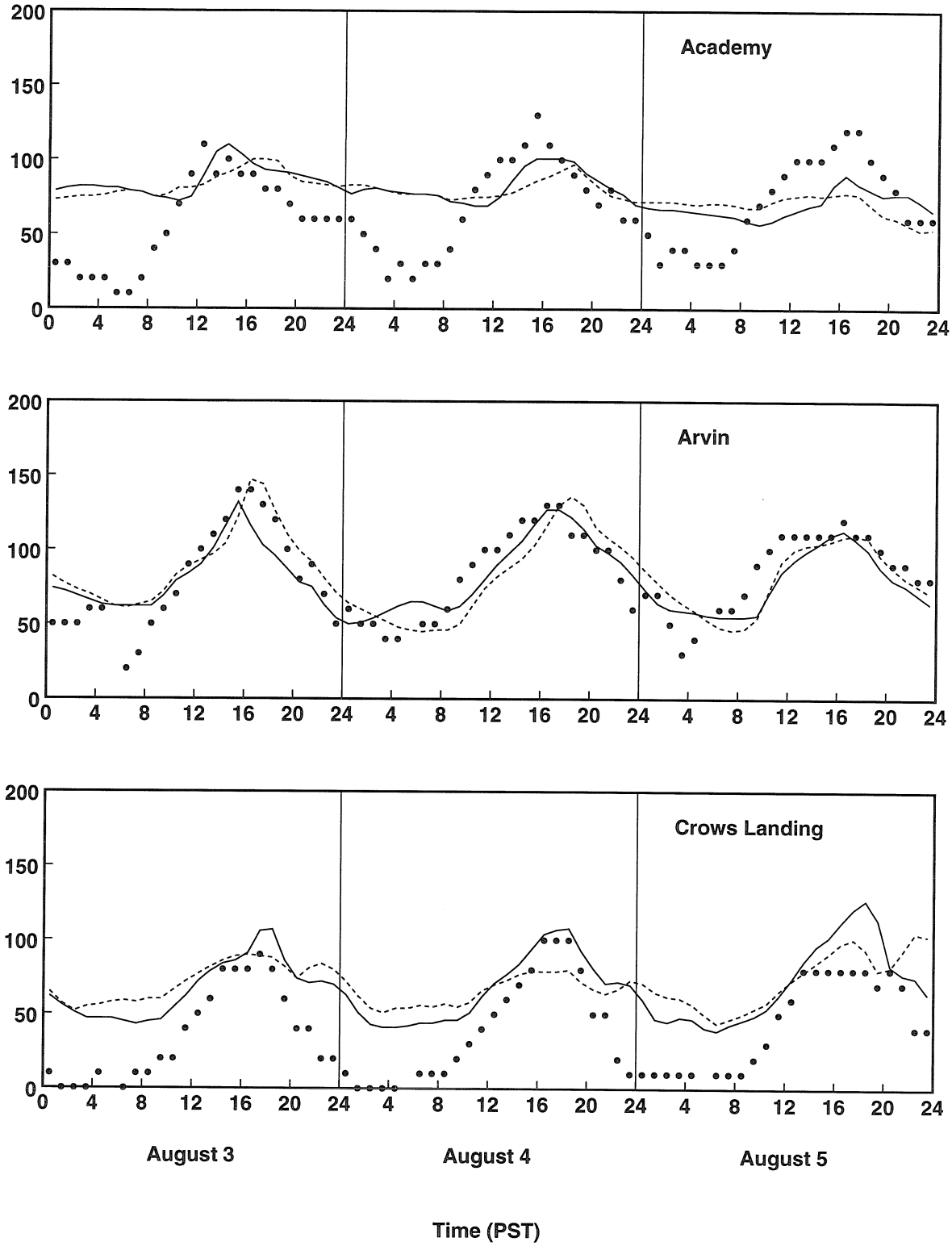


Figure 5.29: Time series plot of observed ozone concentrations in ppb (solid circles), base case ozone predictions (solid line) and base case with wind reduced by a factor of 2, with O3 deposition, SO2 and PAR boundary conditions reduced by a factor of 10 and 5 respectively (dashed line).

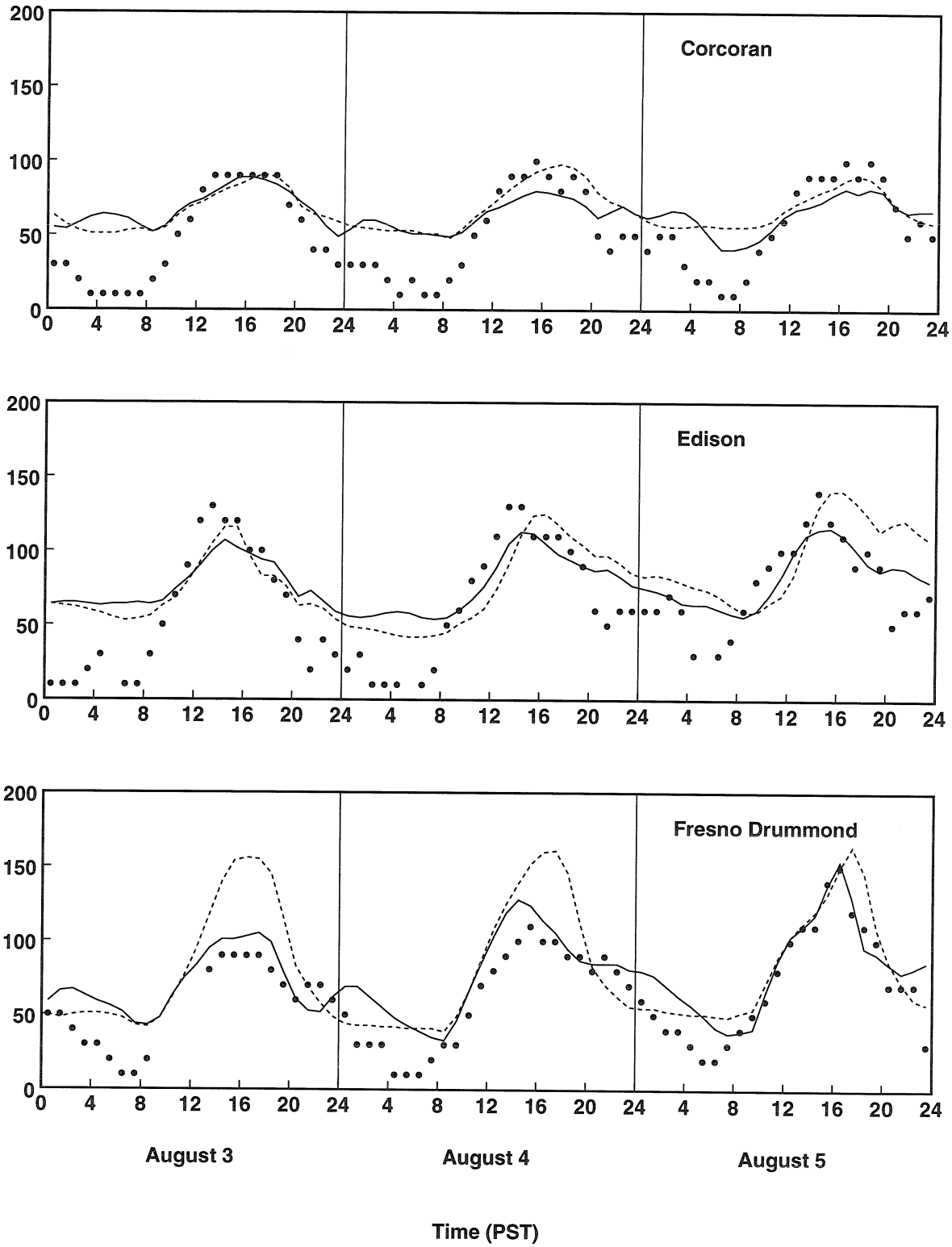


Figure 5.29: (Continued)

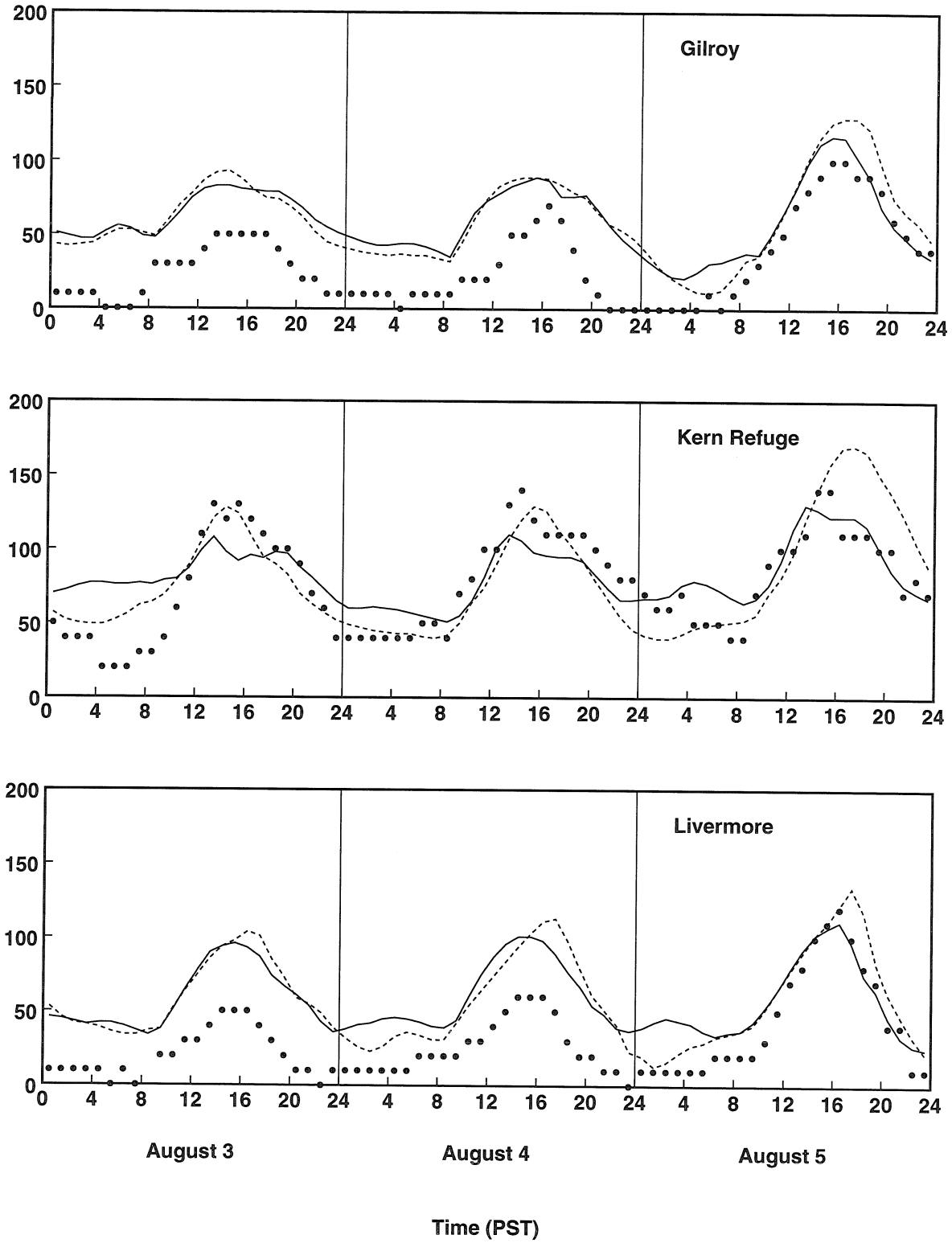


Figure 5.29: (Continued)

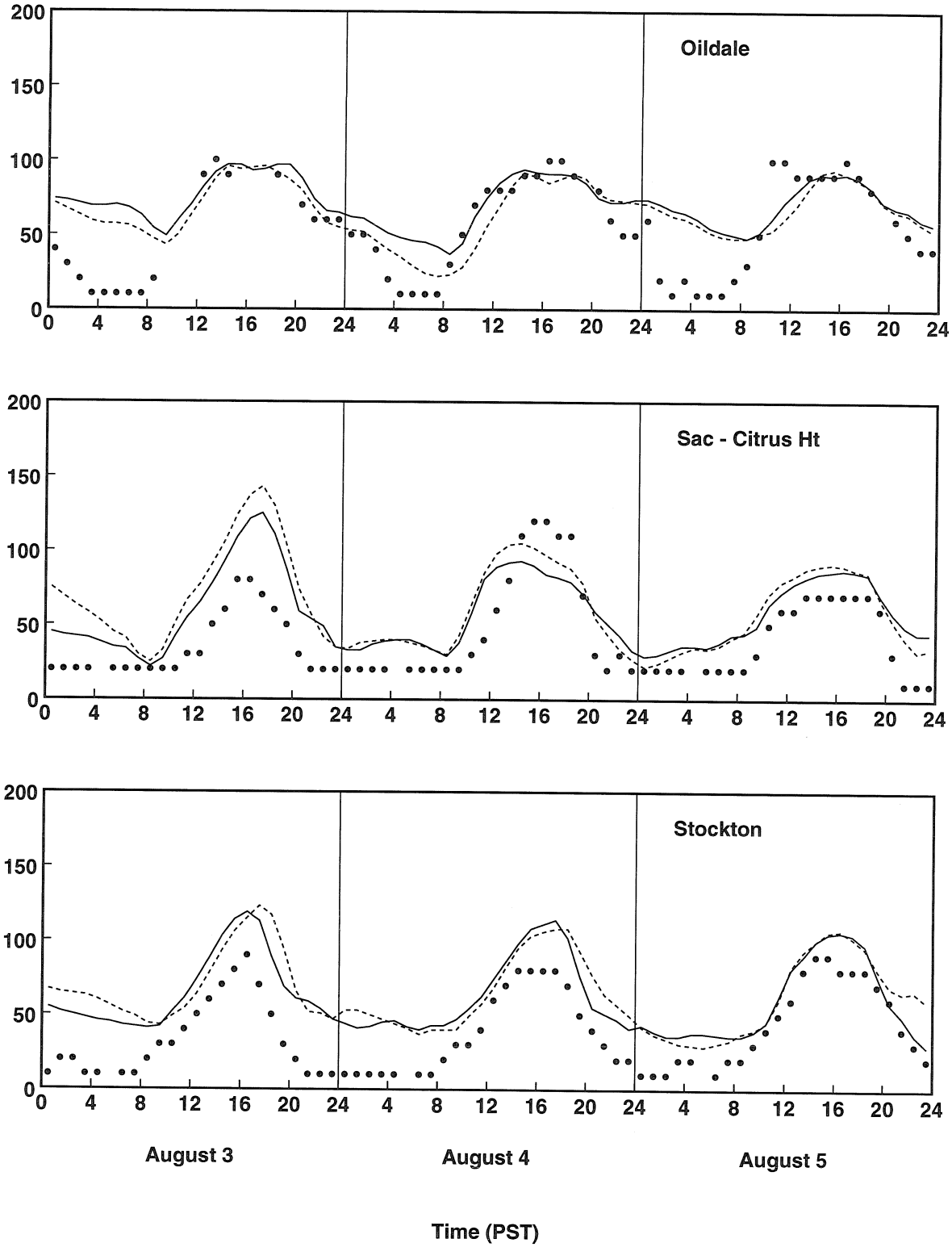


Figure 5.29: (Continued)

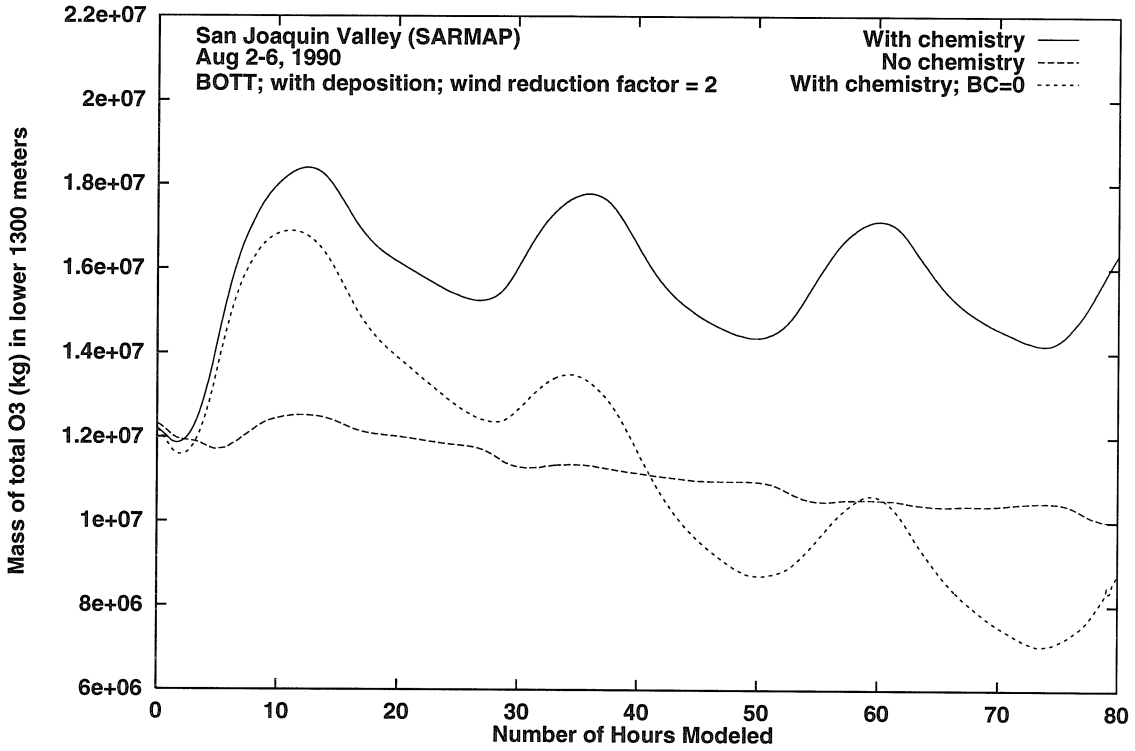


Figure 5.30: Total ozone mass in the lower 1,300 m of the San Joaquin Valley predicted by the SARMAP model with a wind factor reduction of 2, SO2 boundary conditions reduced by a factor of 10, and PAR boundary conditions reduced by a factor of 5. The run starts at 0500 hrs of August 2, 1990.

5.5.3 Zero Emissions

The zero emissions simulation consists of reducing all species emissions to zero in all grid cells and at all times during the episode. The comparison of the zero emissions run with those of the base case run is to ensure that the model responds appropriately to radical emission changes. Physically, upon zeroing the emissions one would expect the level of predicted concentrations to approach a level determined by the inflow conditions. Such response in the dynamics of the simulation is crucial for the suitability of the model to assess the effectiveness of control strategies designed by the application of photochemical models.

Figure 5.20 compares the results from the zero emissions conditions with the base case run for the August 1990 episode. Emissions are set to zero for all species starting on the model spin-up on August 2. Most stations do not exhibit the strong temporal variations that are associated with emissions. As expected, stations near the boundary are less sensitive to emission perturbations. Crows Landing, Corcoran, Gilroy, Livermore, and Stockton exhibit a small decrease in peak ozone concentrations. These stations are influenced by the high background ozone levels imposed by boundary conditions (40-50 ppb in the lower grid cells as shown in Figure 5.19). Such high background concentrations are also the cause of overpredicting ozone levels at night even when the emissions have been zeroed. These stations are the same sites that present the greatest sensitivity to boundary conditions as shown in Figure 5.19. As a result, it is recommended that the values of boundary concentrations, in particular those used in the eastern part, be re-examined. Inland sites, however, exhibit the expected strong sensitivity to emission reduction. For instance, peak predicted ozone concentration drops as much as 66% at locations like Fresno and Edison during the zero emissions simulation. In addition, ozone concentrations for the zero emission case are relatively constant at inland sites. Such behavior is the effect of constant boundary conditions assumed for the modeling of the episode.

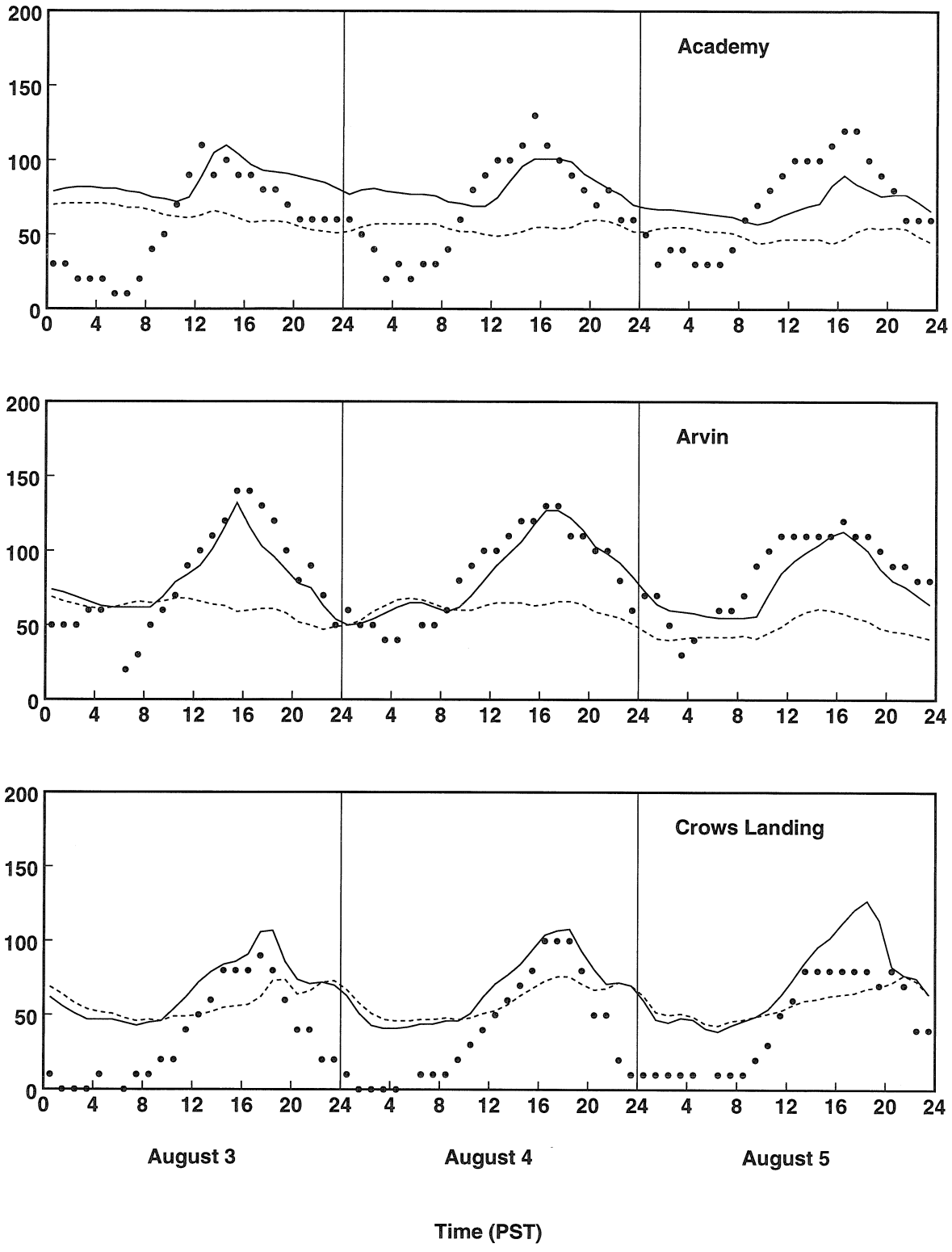


Figure 5.31: Time series plot of observed ozone concentrations in ppb (solid circles) and base case predictions (solid line) and zero emissions ozone predictions (dashed line).

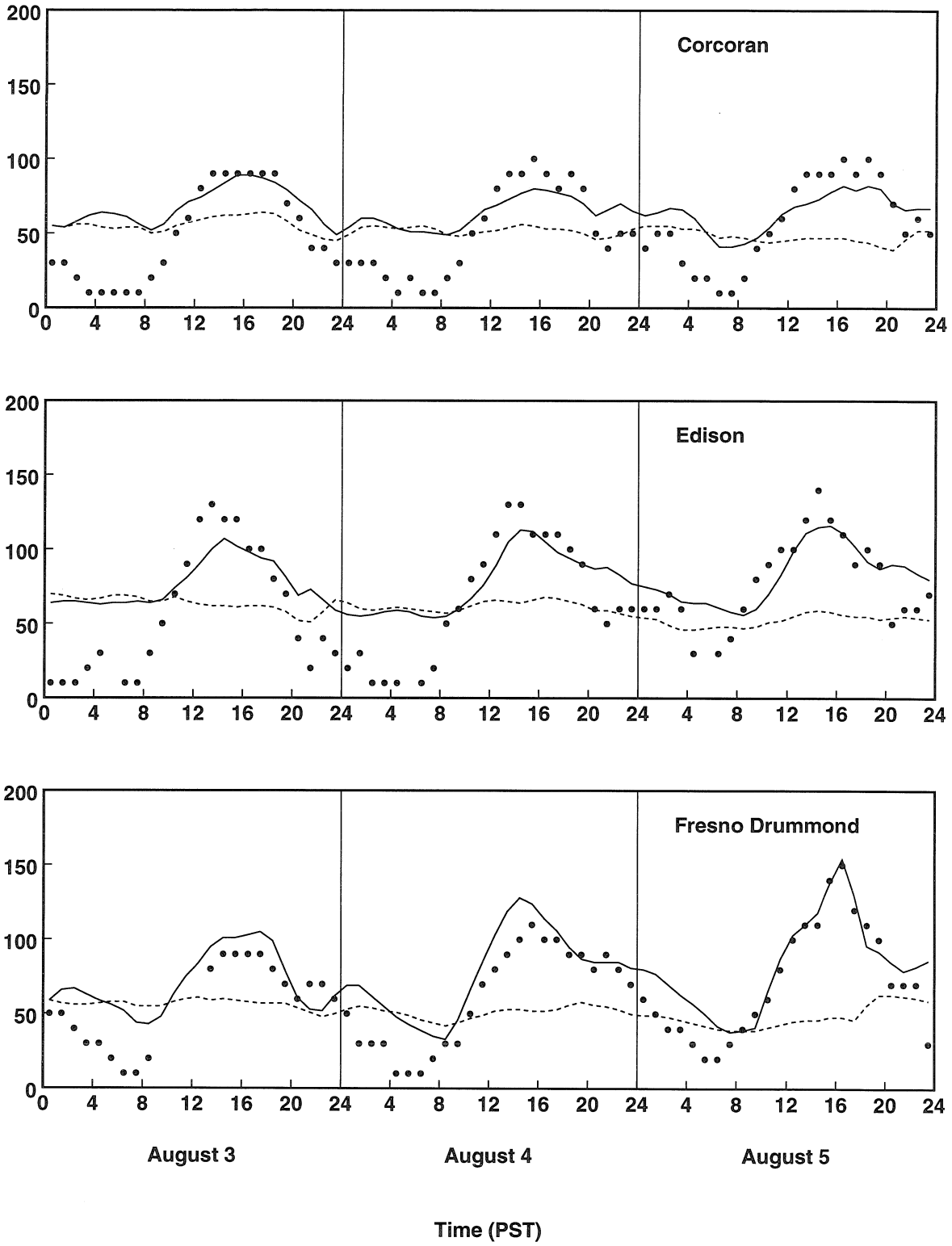


Figure 5.31: (Continued)

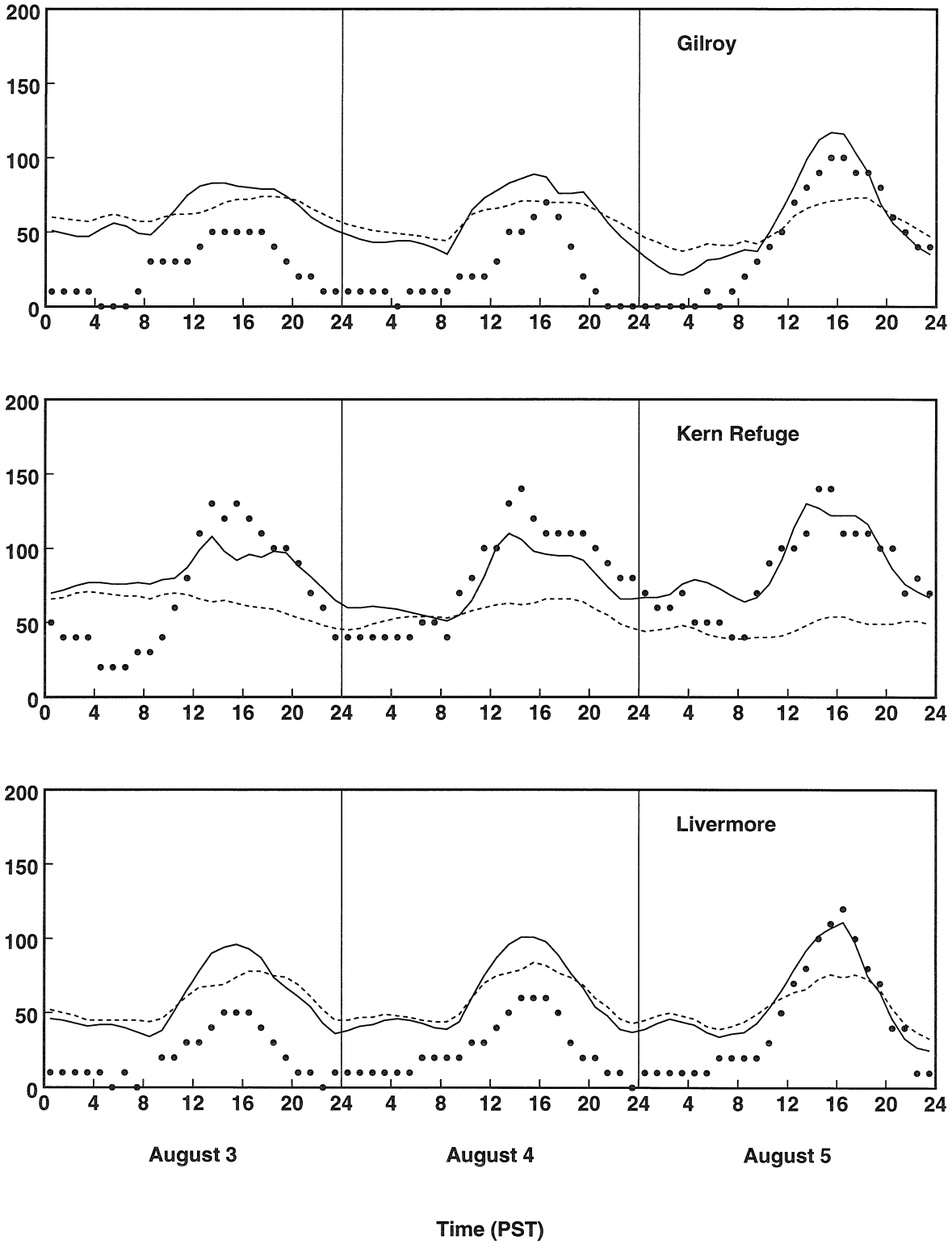


Figure 5.31: (Continued)

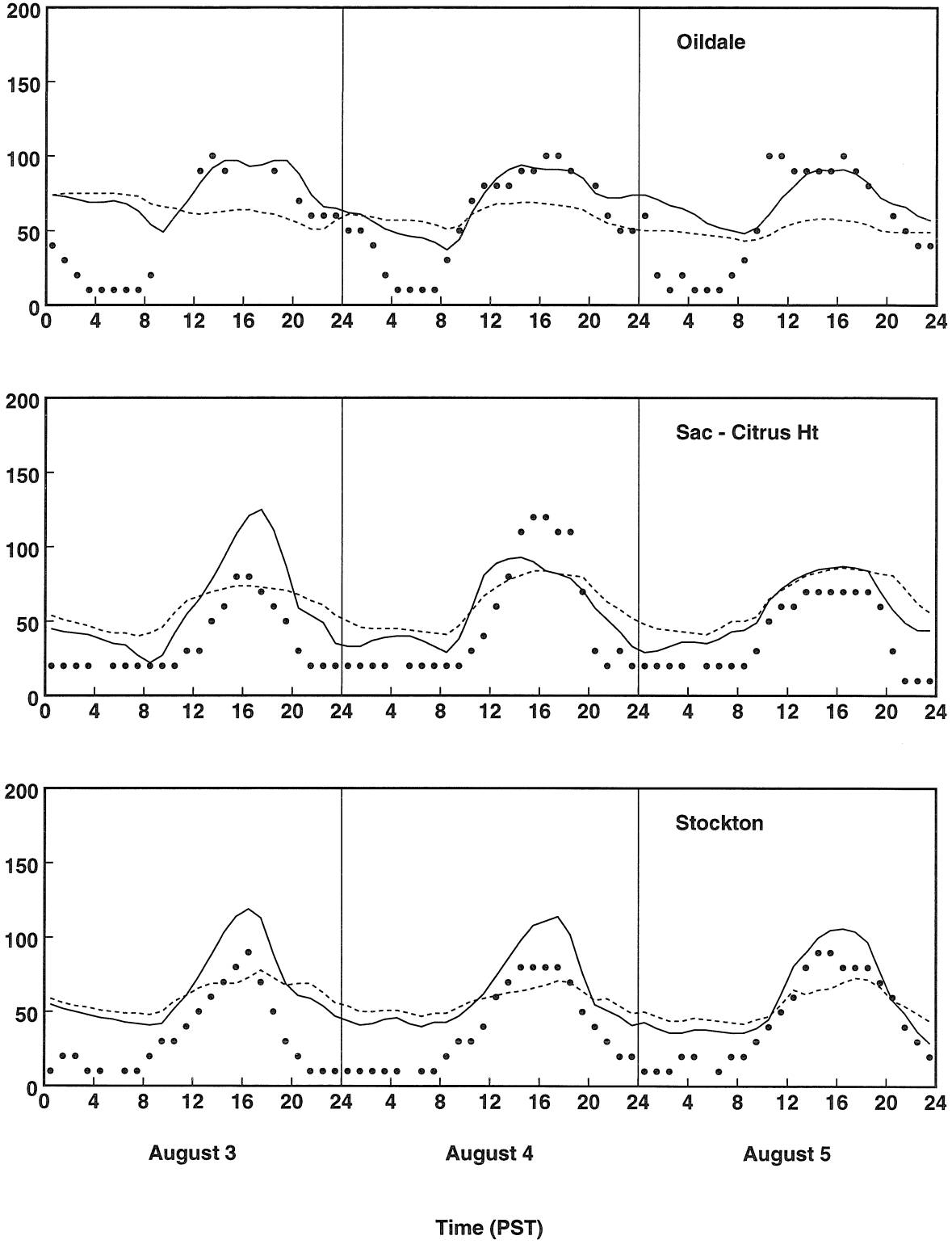


Figure 5.31: (Continued)

5.5.4 Zero Dry Deposition

The zero deposition simulation consists of setting deposition velocities for all the species to zero at all times. Comparison of zero deposition predictions with those of the base case addresses the effect of dry surface deposition removal on concentrations of primary and secondary species. For primary species (NO, NO2, and VOC's) it is expected that downwind concentrations are higher than those of the base case when deposition is neglected. For secondary species, such as O3, the changes in concentration are dependent on the propagation of primary species changes through the nonlinear chemistry and ozone deposition itself. Figure 5.21 shows that predicted ozone concentrations increased throughout the valley. Locations near the coast exhibit an ozone increase of about 10 ppb. Ozone concentration at inland sites increased by 20-40 ppb. The farthest downwind sites are most sensitive to perturbations of deposition velocities.

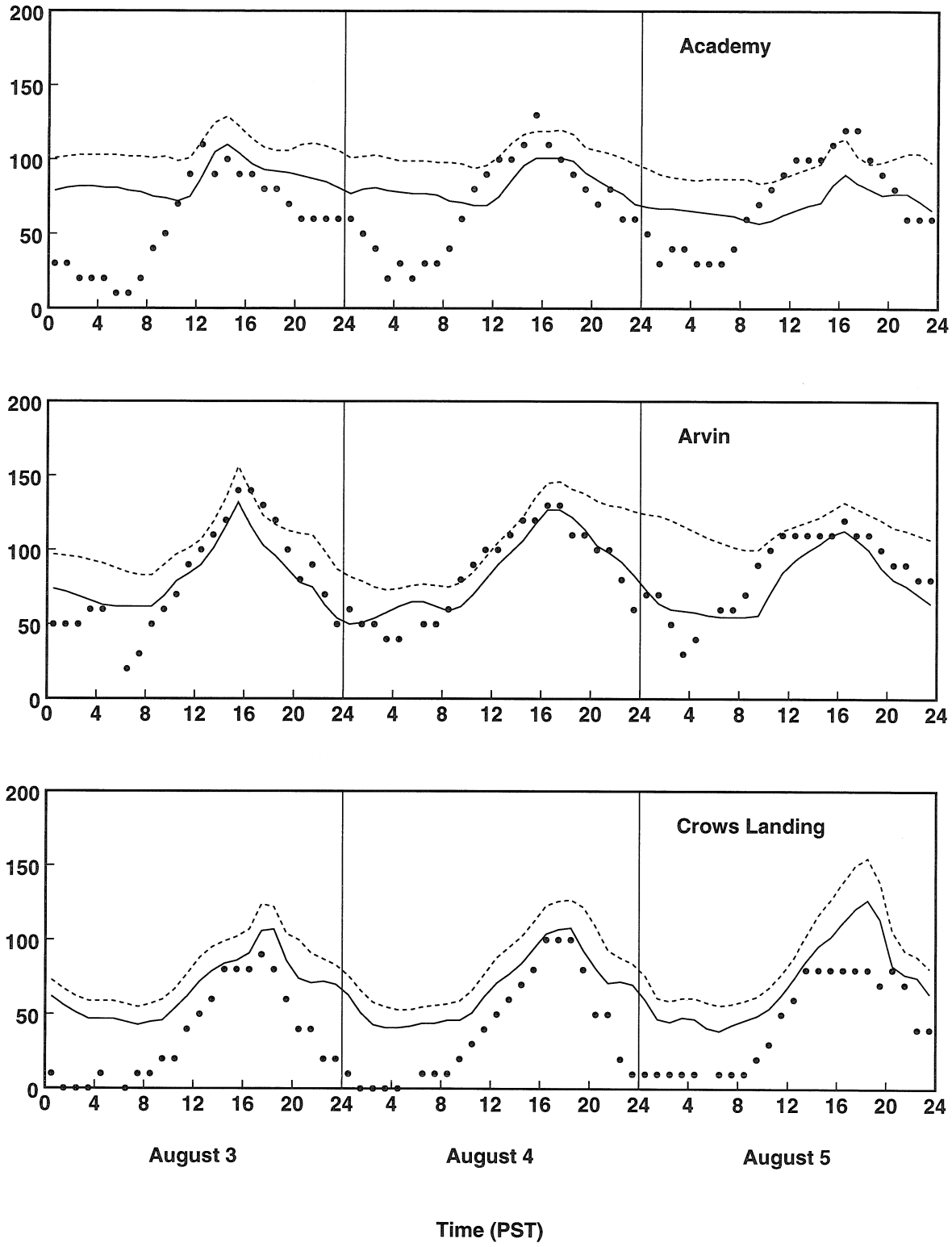


Figure 5.32: Time series plot of observed ozone concentrations in ppb (solid circles) and base case predictions (solid line) and zero dry deposition ozone predictions (dashed line).

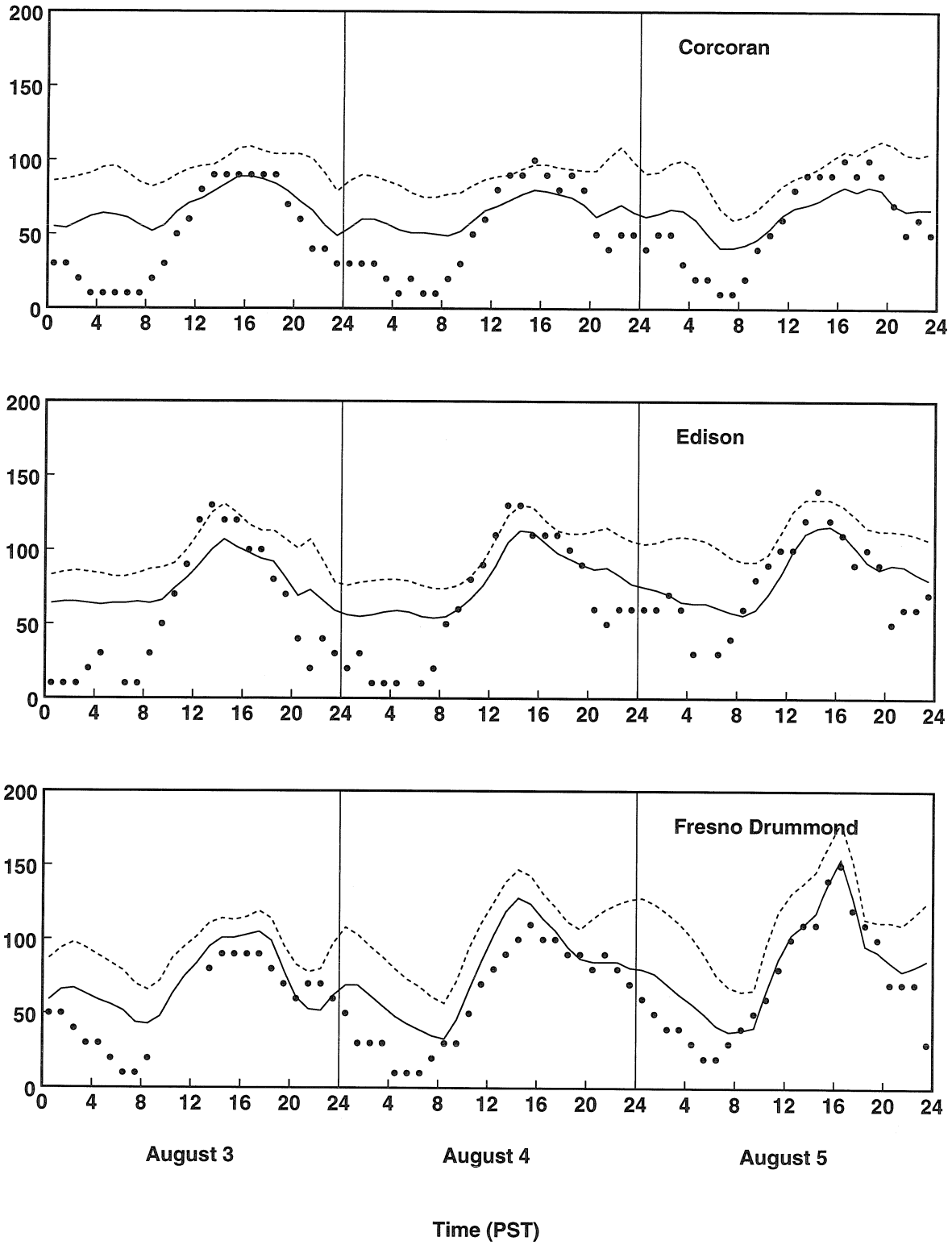


Figure 5.32: (Continued)

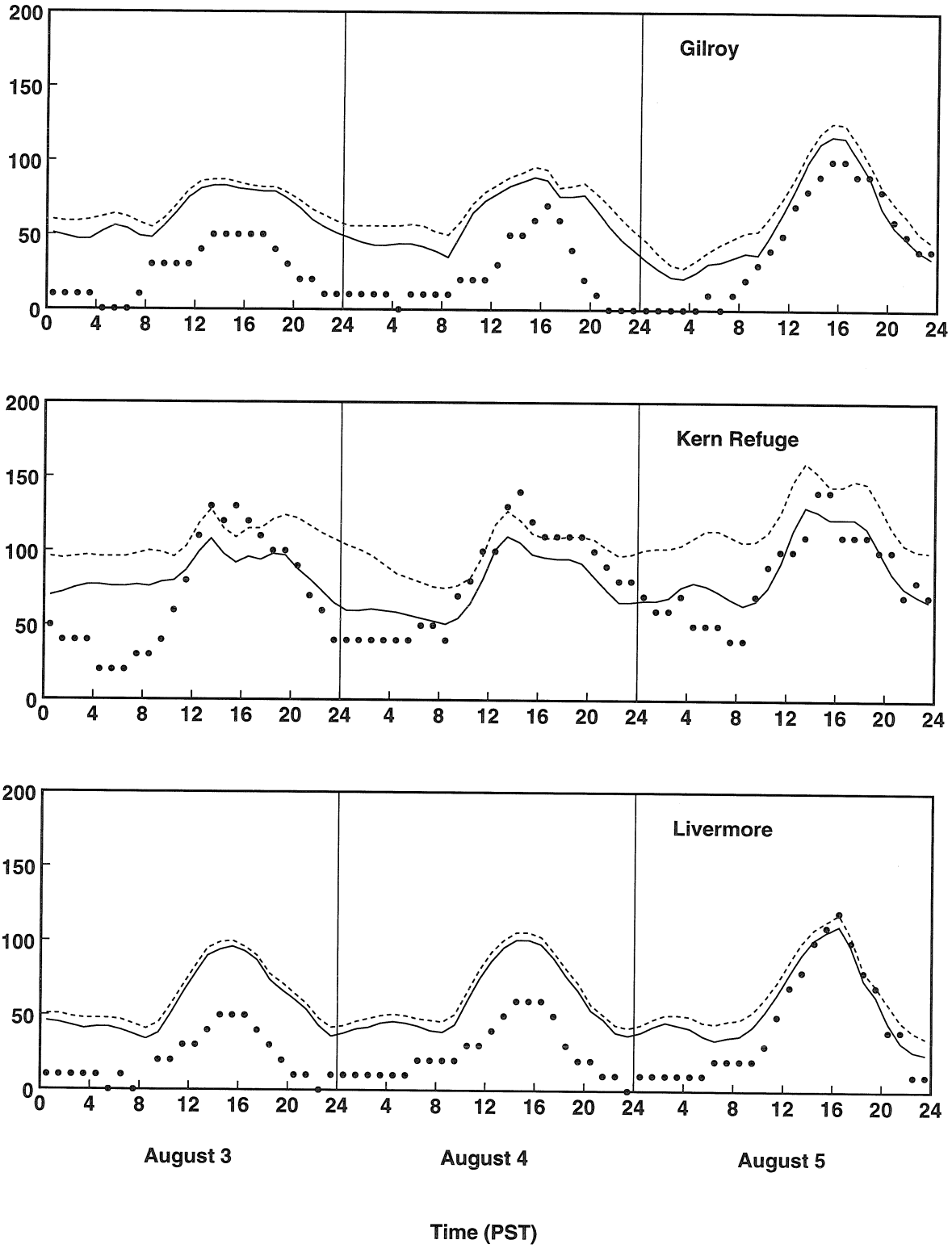


Figure 5.32: (Continued)

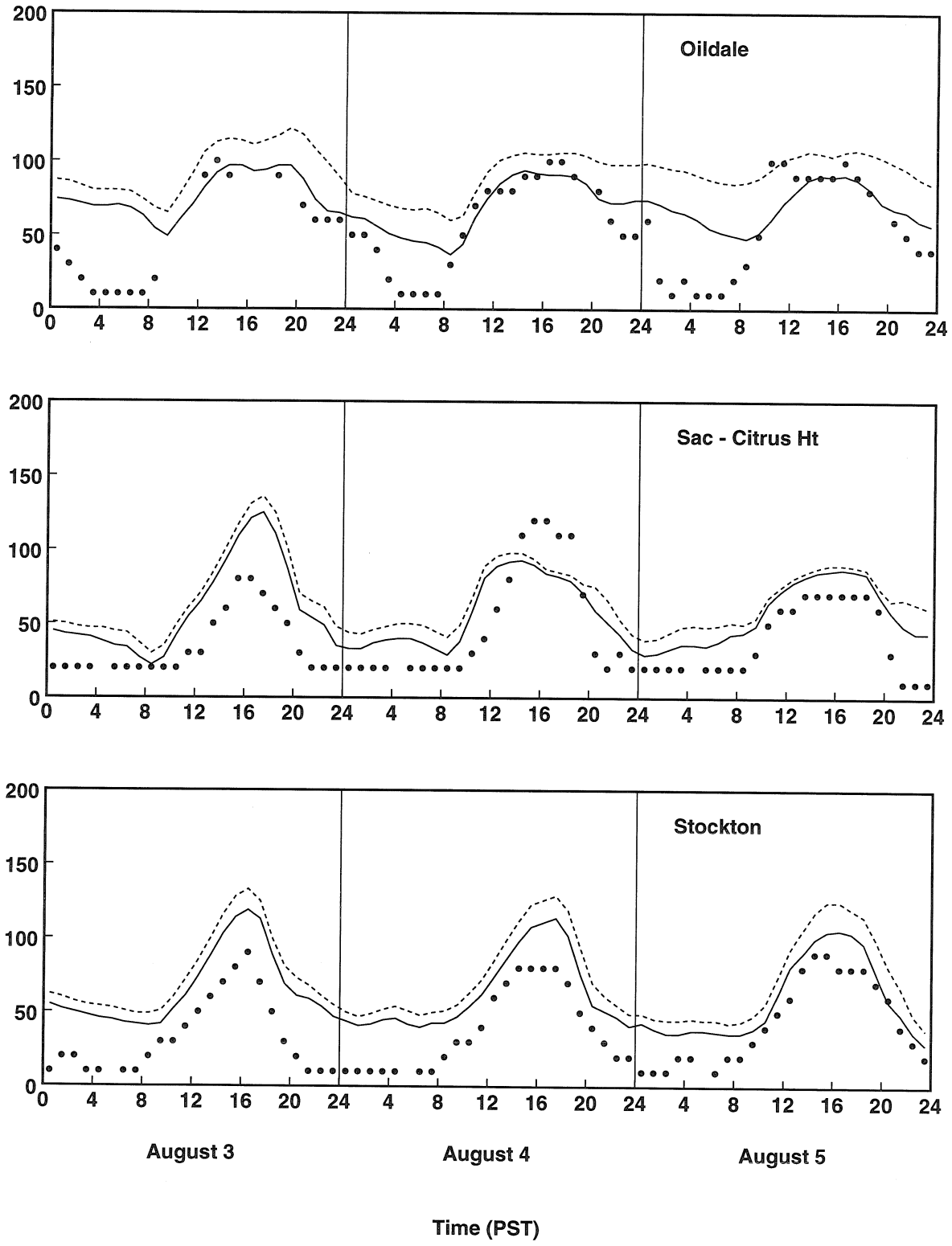


Figure 5.32: (Continued)

References

Blumenthal D. L. (1993) Field Study Plan for the San Joaquin Valley Air Quality Study (SJVAQS) and the Atmospheric Utility Signatures, Predictions, and Experiments (AUSPEX) Program. Final report to California Air Resources Board, STI-98020-1241-FR.

Bott A. (1989a) A positive definite advection scheme obtained by nonlinear renormalization of the advective fluxes. *Mon. Weather Rev.* **117,** 1006-1015.

Bott A. (1989b) Reply *Mon. Wea. Rev.* **117,** 2633-2636.

Chang J. S., Brost R. A., Isaksen A., Madronich S., Middleton P., Stockwell W. R. and Walcek C. J. (1987) A three-dimensional Eulerian acid deposition model: physical concepts and formation. *J. Geophys. Res.* **92,** 14681-14700.

Crowley, W. P. (1968) Numerical advection experiments. *Mon. Weather Rev.* **96,** 1-11.

Dabdub D. and Seinfeld J. H. (1994) Numerical advective schemes used in air quality models—Sequential and parallel implementation *Atmospheric Environment* **28,** (1994) 3369-3385.

Dyer A. J. (1974) A review of flux-profile relationships. *Boundary-Layer Met.* **7,** 363-372.

Grell G. A., Dudhia J. and Stauffer D. R. (1993) A Description of the Fifth-generation Penn State/NCAR Mesoscale Model (MM5). NCAR Tech. Note, NCAR/TN-398+1A.

Hansen J. E. and Travis L. D. (1974) Light scattering in planetary atmospheres. *Space Sci. Rev.* **16,** 527-610.

Joseph J. H., Wiscombe W. J. and Weinman J. A. (1976) The delta-Eddington approximation for radiative flux transfer. *J. Atmos. Sci.* **33,** 2452-2458.

McRae G. J., Goodin W. R. and Seinfeld J. H. (1982) Numerical solution of the atmospheric diffusion equation for chemically reacting flows. *J. comp. Phys.* **45,** 1-42.

Seaman N. L., Stauffer D. R. and Lario A. M. (1995) A Multiscale Four Dimensional Data Assimilation Applied in the San Joaquin Valley during SARMAP. Part I: Modeling Design and Basic Performance Characteristics. *J. Applied Met.* **34,** 1739-1761.

Sheih C. M., Wesely M. L. and Hicks B. B. (1979) Estimated dry deposition velocities over the north-eastern United States. *Atmospheric Environment,* **13,** 1361-1368.

Tanrikulu S., DaMassa J., and Ranzier A. J. (1996) Photochemical Modeling of August 3-6, 1990 Ozone Episode in Central California Using the SARMAP Air Quality Model. Part I: Model Formulation, Description, and Basic Performance. Preprints, Ninth Joint Conference on Applications of Air Pollution Meteorology with AWMA, Atlanta, Georgia, AMS, January, 28- February 2, 1996.

Walcek C. J., Brost R. A., Chang J. S. and Wesely M. L. (1986) SO_2 and HNO_3 deposition velocities computed using regional landuse and meteorological data. *Atmospheric Environment,* **20,** 949-964.

Walcek C. J. and Taylor G. R (1986) A theoretical method for computing vertical distributions of acidity and sulfate production within cumulus clouds. *J. of the Atmosph. Sci.* **43,** 339-355.

Wesely M. L and Hicks B. B. (1977) Some factors that affect the deposition rates of sulfur dioxide and similar gases on vegetation. *J. Air Pollut. Cont. Assoc.* **27,** 1110-1116.

Yanenko N. N. (1971) *The Method of Fractional Steps.* Springer, New York.