VENTILATION SYSTEM PERFORMANCE

11th AIVC Conference, Belgirate, Italy
18-21 September, 1990

Paper 13

STRUCTURE OF MODELS FOR THE PREDICTION OF AIR FLOW AND CONTAMINANT DISPERSAL IN BUILDINGS

Richard A. Grot[1], James A. Axley[2]

1
Lagus Applied Technology
Olney, MD 20832
USA

2
Building Technology Program
Massachusetts Institute of Technology
Cambridge, MA 01273
USA

Abstract

This paper treats the structure of models for predicting interzonal airflow and contaminant dispersal in buildings. It will discuss the mathematical structure of such models, the use of modern data structures, the application of structured program techniques and the use of object-oriented structures for the development of users interfaces and building description processes. Two computer models developed at National Institute of Standards and Technology (NIST) will be used as examples of how these techniques can be applied to air flow analysis and contaminant dispersal: **NBSAVIS** - a building description processor for multizone buildings, and **CONTAM88** - an interzonal airflow and contaminant dispersal analysis program. **NBSAVIS** treats the building as a collection of physical objects which have flow and leakage characteristics and creates a building idealization (flow paths, zones and contaminant source data) from the physical description of the building. It will be demonstrated that with the proper data structure, general interfaces which are both user-friendly and physically correct can be developed. **CONTAM88** combines previously developed interzonal airflow and contaminant dispersal analysis programs **AIRMOV**, **CONTAM86** and **CONTAM87** into one model which includes such features as contaminant reactions, pressure induced contaminant sources for modeling radon entry and plateout and disposition of contaminants onto surfaces. An example is presented on the use of these programs for airflow and contaminant dispersal analysis in a large apartment building.

1 Introduction

The National Institute of Standards and Technology (NIST) (formerly the National Bureau of Standards (NBS)) has over the past several years developed a series of computer models for calculating air flow and contaminant dispersal in multizone buildings.[1-5] Though these models have been developed with the purpose of providing state-of-the-art, documented, public domain algorithms for calculating interzonal air flows and contaminant levels in buildings, they have required a rather laborious preparation of input data and sophisticated building idealization in order to model realistic building configurations. In an attempt to facilitate the use of these models in the analysis of indoor air quality problems in buildings, a preprocessor program **NBSAVIS** (The National Bureau of Standards Air Infiltration, Ventilation and Indoor Air Quality Analysis Program in Multizone Buildings) has been developed which assists in the preparation of input files for air flow and indoor air quality analysis. The main features of the programs **AIRMOV** [1,2] and the programs **CONTAM86** and **CONTAM87** [4,5] have been combined into a program called **CONTAM88** which integrates multizone air flow analysis and contaminant dispersal analysis into one program.

The program **NBSAVIS** is a building description processor for multizone buildings. It is menu driven and has the capability of editing and creating a building description and calculating the required zone and opening data for a multizone air flow and contaminant dispersal analysis.

The steps in performing an air movement and indoor air quality analysis of a building using the above mentioned programs are:

Create and edit a building description using **NBSAVIS**

Perform and air movement analysis and contaminant dispersal analysis using **CONTAM88**

If a more detailed indoor air quality analysis is desired use **CONTAM86** or **CONTAM87**

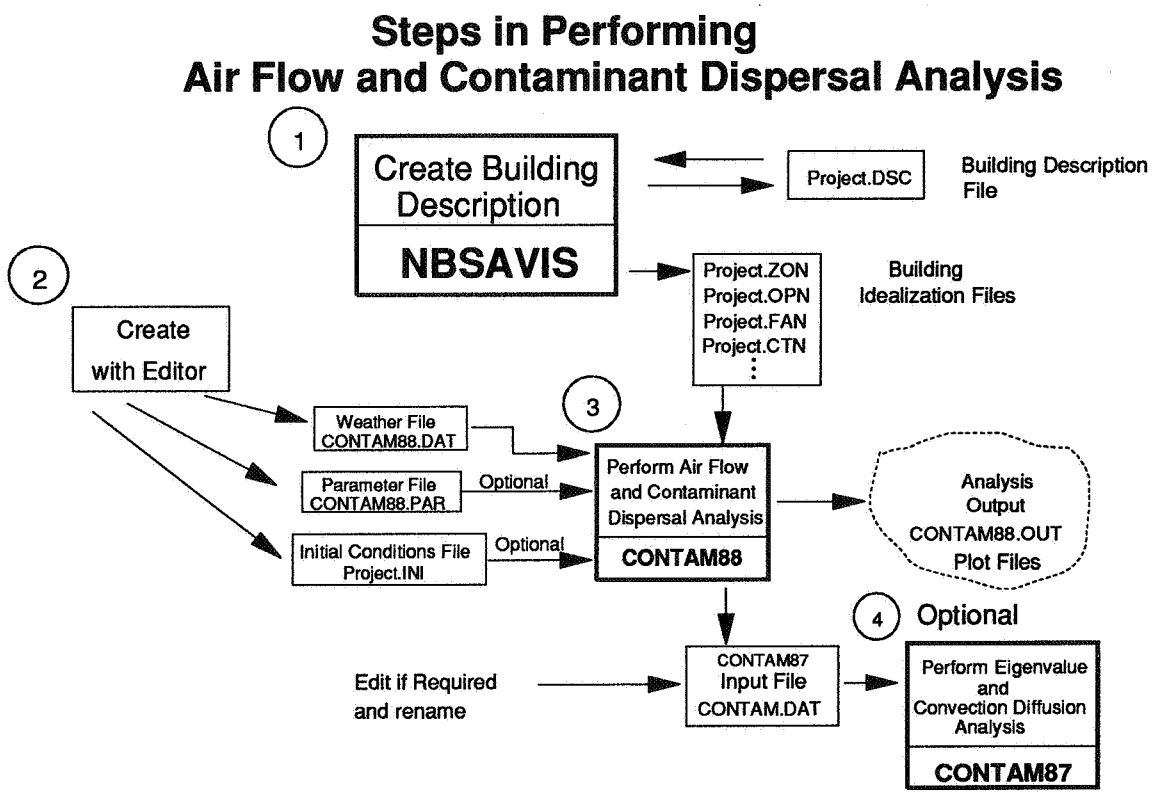


Figure 1. Steps in Performing Air Flow and Contaminant Analysis

2 Description of the Program NBSAVIS

The program **NBSAVIS** is designed to run as a preprocessor for the interzonal air movement and contaminant dispersal analysis program **CONTAM88** and the indoor air quality analysis program **CONTAM87**. It takes data on the building envelope construction, internal configuration, HVAC system and contaminant generation sources and prepares input files for **CONTAM88**. **NBSAVIS** is designed to create a building description which is modular in nature and the description of the building can be refined as more data are acquired or as a more detailed analysis of the air flows and contaminant dispersal into and within the building is desired. **NBSAVIS** supports the following basic functions:

1.) Creation of a new building description.

2.) Editing of an existing building description.

3.) Saving a created or edited building description to disk.

4.) Calculation of the opening, zone and fan data from the building description and preparation of the input files for **CONTAM88**.

In addition **NBSAVIS** has two program utility functions which allow:

5.) Changing certain program parameters such as the physical units used and how building leakages are described (either as leakage area or flow coefficients).

6.) Returning to DOS in an orderly manner.

The input of data for the program **NBSAVIS** is controlled by a series of screen originated sub-routines for the description of the building, the types of exterior walls, exterior doors, windows, interior walls, interior doors, open passageways, miscellaneous openings, filters and fans, descriptions of the rooms of the building and descriptions of the building HVAC systems.

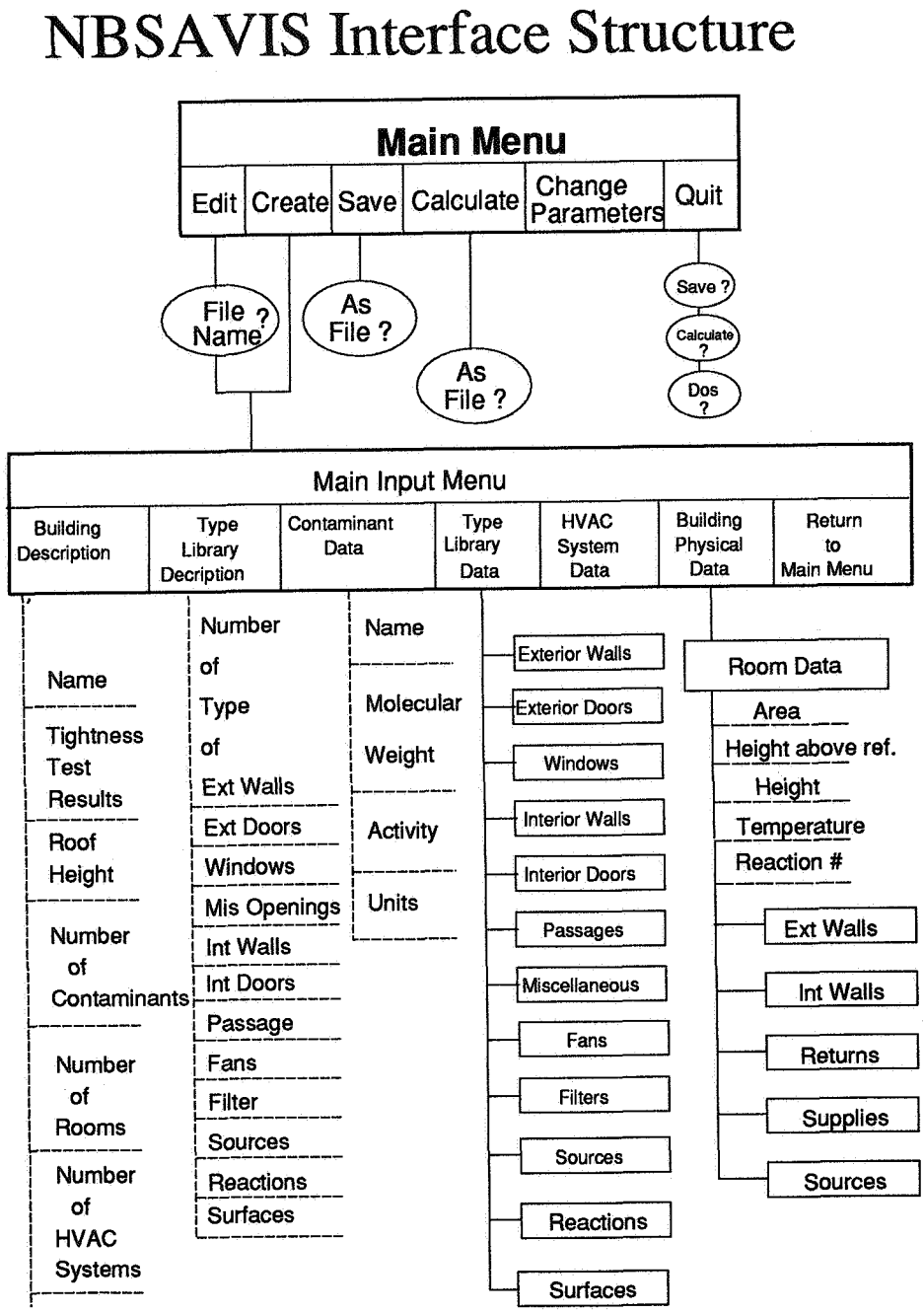


Figure 2. **NBSAVIS** Interface Structure

For **NBSAVIS** a building consists of rooms and HVAC systems. Rooms for **NBSAVIS** can be actual building rooms, hallways, basements, garages, porches, stairwells, elevator shafts, return air shafts, exhaust shafts, supply air shafts and in general any physical volume. Rooms have physical properties, contaminant sources, HVAC system connections, exterior walls and interior walls. Exterior walls, interior walls and HVAC connections contain the flow paths through which air and contaminants can move into, out of and throughout a building. Exterior walls consist of an opaque wall area, exterior doors, vents, windows, intake fans and exhaust fans. Interior walls consist of an opaque wall area, interior doors, open passages, vents, exhaust fans and intake fans. An exterior wall points to the outside, an interior wall points to a previously defined adjacent room.

There is no restriction as to the size of the building which **NBSAVIS** can handle. Features of large buildings such as stairwells, elevator shafts, exhaust shafts, return air shafts and supply shafts can be modeled using **NBSAVIS** and **CONTAM88** by considering such features as rooms. The author has successfully modeled a twelve story apartment building with two stairwells, a vent shaft, an elevator shaft as a 100 zone building using **NBSAVIS** and **CONTAM88** without taxing the resources of 640K PC.

The specific leakage data, contaminant source data and fan flow rates are specified in a library of building component types. The building component types that **NBSAVIS** support are exterior wall types, exterior door types, window types, interior wall types, interior door types, open passage types, miscellaneous or vent types, fan types, filter types and contaminant source types. The actual building part is described by specifying what types it consists of. For example the north exterior wall of the master bedroom is of exterior wall construction type # 1, has one exterior door of exterior door type # 3, two exterior windows of window type # 1, no vents, no exhaust fans and no intake fans. The master bedroom has two contaminant sources, one of type #2 which is a kerosene heater and another of type # 3 which is an air freshener. Given such information on the construction of the building and the location of contaminant sources, **NBSAVIS** can construct the idealization of the building in terms of flow paths, zones and contaminant generation rates required to perform airflow and contaminant dispersal analysis.

2.1 Type Component Library Data

The program **NBSAVIS** uses a building specific library of leakage types, filters and fans which exist in the building. The purpose of this library is to allow the same building component to be used in various parts of the building without repeatedly inputting the data which describes the component. The type library contains the physical and leakage data for exterior walls, exterior doors, windows, interior walls, interior doors, open passages (large openings), miscellaneous leakages (vents), fans, contaminant filters, contaminant sources, kinetic reactions, and exterior surfaces. The description of the buidling layout is speicified in terms of the types contamined in the type library. One only needs to specify the type number of the component and number of components which exist in the building part for which one is preparing a description. For example the family room of the building has an exterior north facing wall with a floor height of 2.5 meters, a width of 4 meters of exterior wall type # 2, this wall has 2 exterior windows of type 4 and 1 exterior door of type 4.

2.2 Physical Building Data

The physical building data are the data required to describe a building in terms of its physical components. For a small building the building is considered to be composed of rooms and HVAC systems. The rooms and HVAC systems are the well-mixed zones of the flow and contaminant dispersal analysis idealization which **NBSAVIS** creates. Attics, basements,

halls, stairways, garages, porches etc are considered to be rooms. Rooms are connected to the HVAC systems through room supplies and returns, to other rooms through interior walls and to the exterior through exterior walls. These connections are the flow paths through which contaminants and air move throughout, into and out of the building. Rooms also have contaminant sources and sinks which generate or remove contaminants.

2.3 General Data Structure

In general, data for **NBSAVIS** are organized as either global variables or as a field of a record in a linked list. Global variables are usually properties of the whole building (such as building height, building leakage area) or govern the way one wishes to input data (metric or English units, use leakage area or flow coefficient for leakage properties, use mass flow rates or volumetric flow rates). Global variables always exist and a value must be always specified or the default value is assumed.

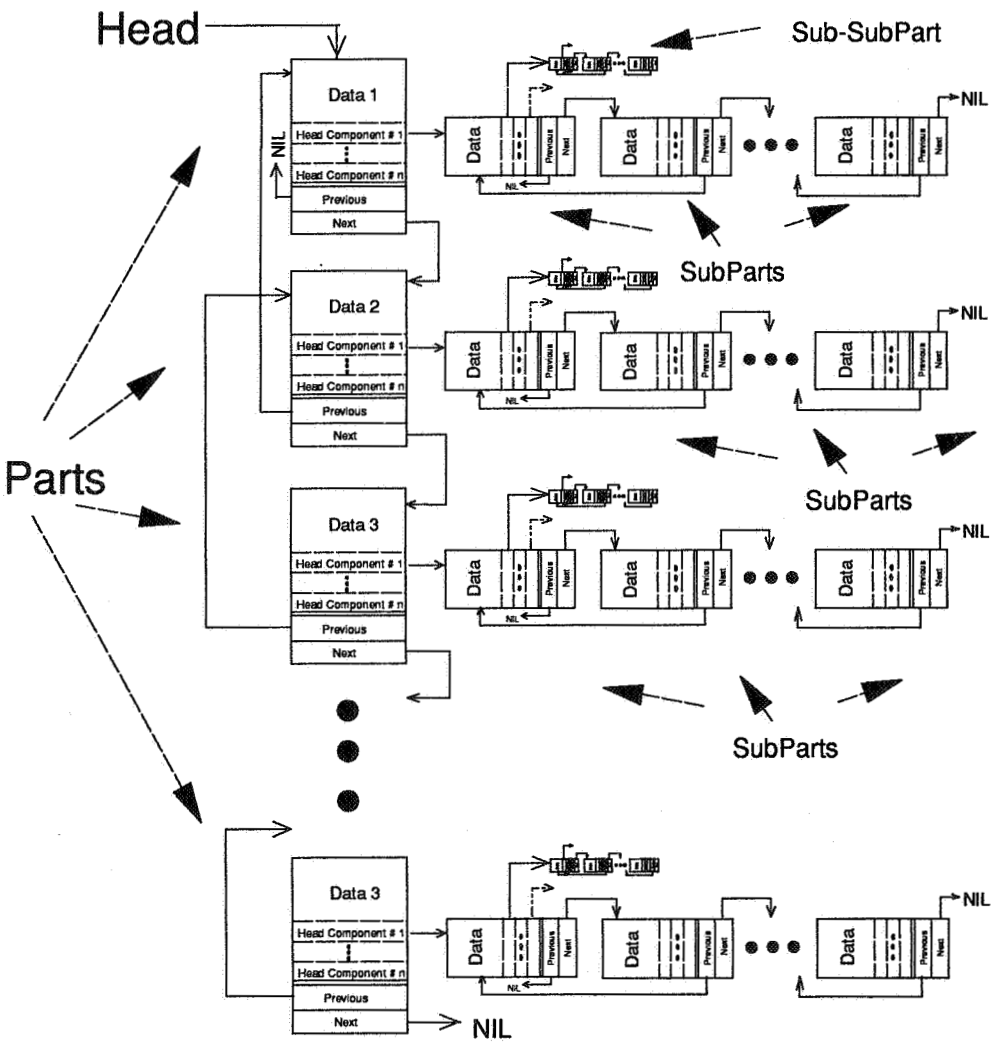


Figure 3. Schematic of Data Structure of NBSAVIS.

Linked lists (**NBSAVIS** uses double linked lists) contain a series of related records (see figure 3). Records in general contain physical properties of item described by the record (for example, a exterior wall type, a room, etc), pointers to the next and previous record in the list and pointers to the beginning of a list or records describing subparts of the physical item described (for example, the exterior walls of a room, the doors and windows of an exterior wall).

This data structure has been found to be efficient in describing a building in terms of its parts and the relationship between various components and the parts of the building. It allows for easy insertion or deletion of parts and components and permits a description of the interdependencies among the parts and components. This data structure also corresponds much to the way one physically views a building. The division of the data for a building into the data required to specify the physical description of the buidling and its HVAC systems and the data required to describe the flow and contaminant emission characteristics of building conmponents allows the modification of one class of data without altering the other class of data. The data for the physical description of the buidling requires only geometrical data and a specification of the components making up the part of the buildings being described. It does not depend on the physical properties of the componet which specifies the components air leakage, air flow or contaminant emission characteristics. Those properties are specified in the component library.

3 Description of CONTAM88

The program **CONTAM88** implements the airflow and contaminant dispersal theory described in the reports and papers of Walton and Axley in one software package. The structure of the program **CONTAM88** is shown in figure 30. **CONTAM88** starts by initializing the global variables and pointers to arrays and link lists. It then checks for the existence of the parameter file. If the parameter file is found in the current directory, the file is open and the parameters are read. If the file does not exist, the default set of parameters is used. **CONTAM88** then opens the weather data file CONTAM88.DTA and reads the starting time/data and the name of the initial building description. It then proceeds to open the corresponding data files for the zone data, contaminant data, opening data, fan data, surface data and kinetic reaction data. In reading these files the corresponding linked lists and arrays are created and the data is storage in these data structures. If an error occurs in reading these files or in allocating the required data storage, an error flag is set and upon exit from the corresponding input routine, the program is terminated with an error message. **CONTAM88** then reads the weather data, advances to the next data record and reads the next time/date. If an end of file is encountered in advancing the data record, the program terminates. The flow system of equations is then formed and solved. If an error occurs in the solution of the flow system, the program terminates with an error message. The logic for solving the flow system of equations is presented in the next subsection. **CONTAM88** then forms and solves the contaminant dispersal system of equations if the number of contaminants is greater than zero. If an error occurs in the solution of the contaminant dispersal system, an error message is given, however the program does not terminate. **CONTAM88** then read the next building description name. If it is different from the previous building description name, the zone data, contaminant data, opening data, fan data, surface data and kinetic data are read for the new description. Some basic checks of the new building description are made to determine if it is consistent with the previous description: that is that the number of zones has not changed and that the number of contaminants are the same. The next weather data are then read and **CONTAM88** proceeds to solve the new flow and contaminant dispersal systems. This process continues till an end of file is encountered on the weather data file. Note that the weather data on

the last record is not processed; therefore if one wishes to analysis the flows and contaminant dispersal for these weather conditions, a dummy record must be added. Note also that if the next time/data is less then the previous time/date, the program will also terminate.

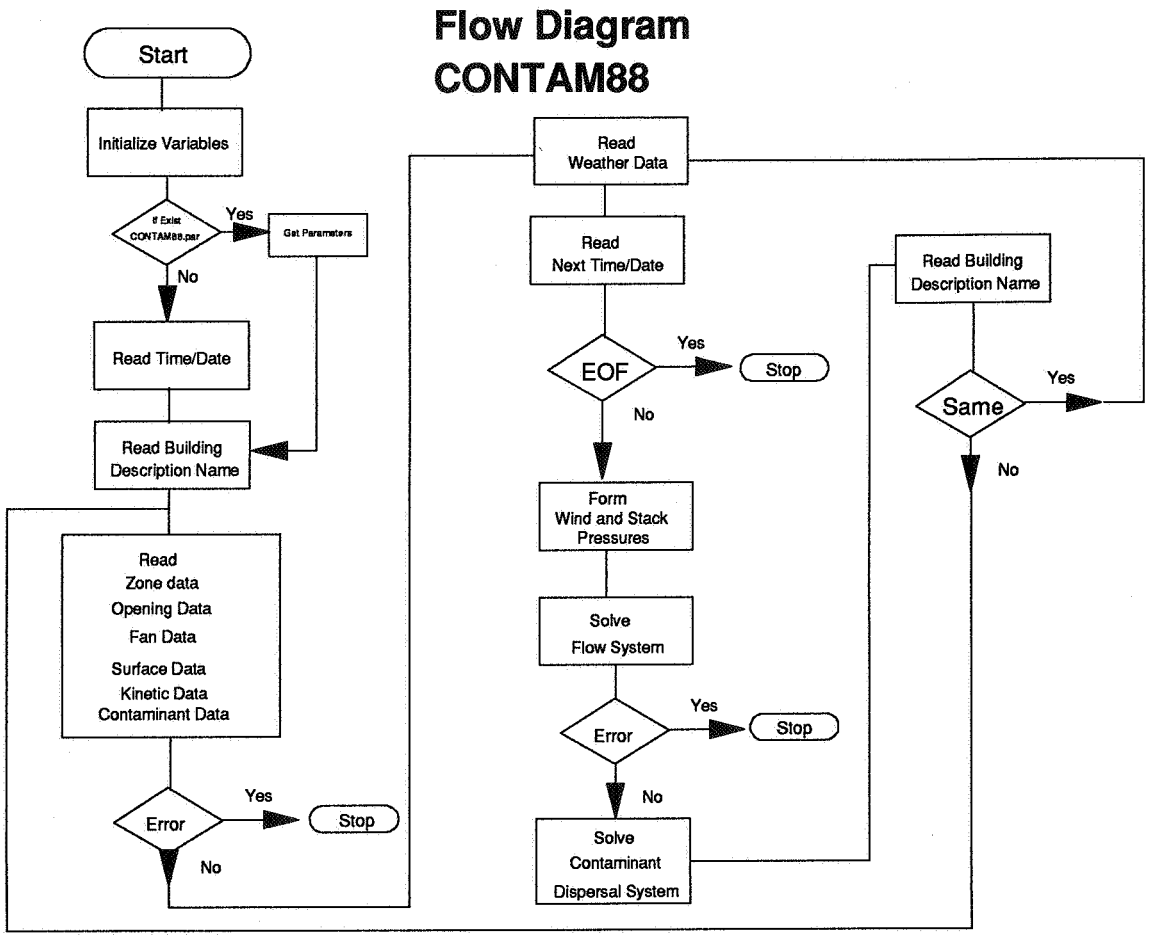


Figure 4. Schematic of Airflow and Contaminant Dispersal Program **CONTAM88.**

The solutions scheme for the air flows in the building is depicted in figure 5. The subroutine solvzp allocates arrays eqn_number and zone_number. If i is an index of a zone, the eqn_number[i] is the equation number of the flow equation for the zone; if n is an equation number, then zone_number[n] is the zone number corresponding to the equation. The system of flow equations is then reordered in such a way that zones which are not coupled by openings (such zones would have a zero diagonal in the Jacobian matrix) to other zones are excluded from the system. The logic used to reorder the flow system equations is shown in figure 6. The diagonal of the Jacobian of the unordered flow system is formed. The equations are renumbered so that all zones with non zero diagonal elements occur first, the zone with zero diagonals in the Jacobians appear last. Examples of zones which would have zero diagonal elements in the Jacobian are HVAC systems or zones coupled by only specified flows. The reorder subroutine returns the number of equations which are to be solved. If this number is zero (as would happen if all flows were specified), the solution algorithm is terminated. One then calculates and stores the square of the air densities in each zone and allocates storage for the bandwidth vector used to determine the storage requirements of the Jacobian matrix. The flow elements are then transversed, deter-

mining their connectivity and their influence on the bandwidth vector. One then determines the sums of all fixed flows into each zone and stores these results in each zone memory structure. Storage is then allocated the element Jacobian matrix, the four vectors need for the solution technique (**B** - the vector of sum of the flows into a zone, **C** - the vector of corrections to the zone pressures, **D** - the previous pressure correction vector, **F** - the vector of the sum of the absolute values of the flows into or out of the zone) and **A** - the Jacobian matrix for the flow system The solution of the flow system is then solved using a Newton-Raphson technique which is described in more detail later.

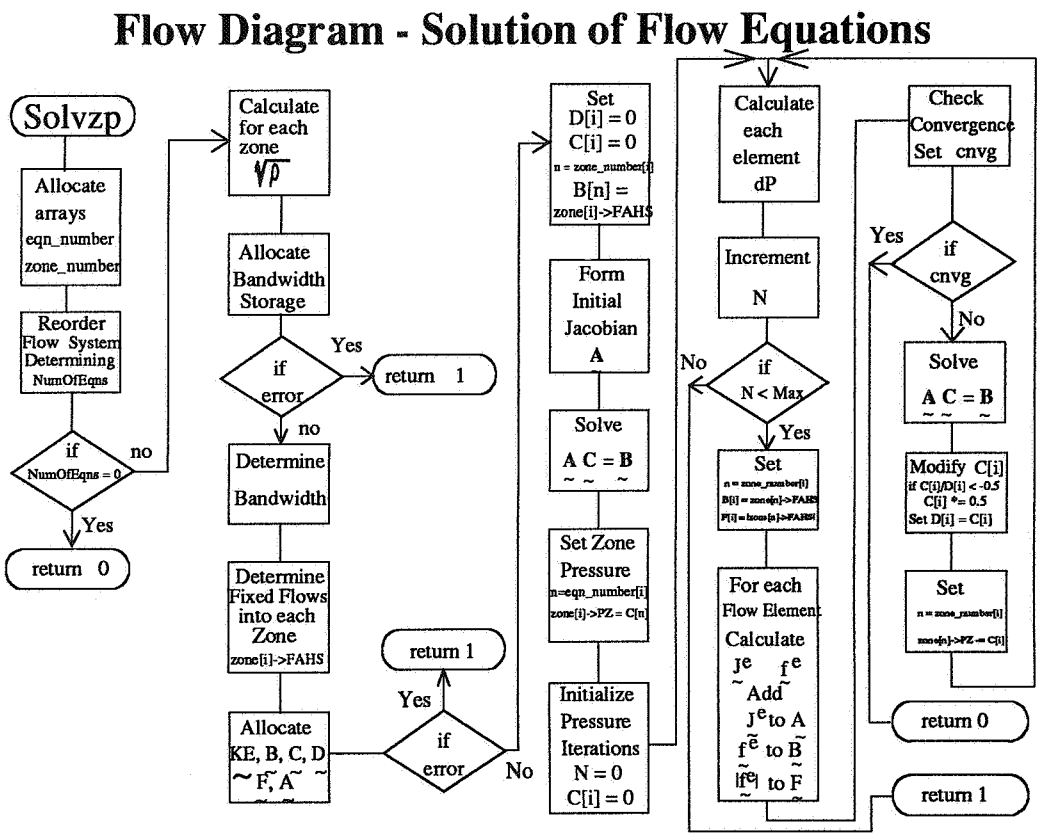


Figure 5. Schematic of Solution Method for Airflow Analysis Used by **CONTAM88.**

After solving for the air flows between zone, CONTAM88 formulates and solves both the steady state and transient contaminant dispersal problem. The general method for solving the contaminant dispersal problem is shown in figure 7.

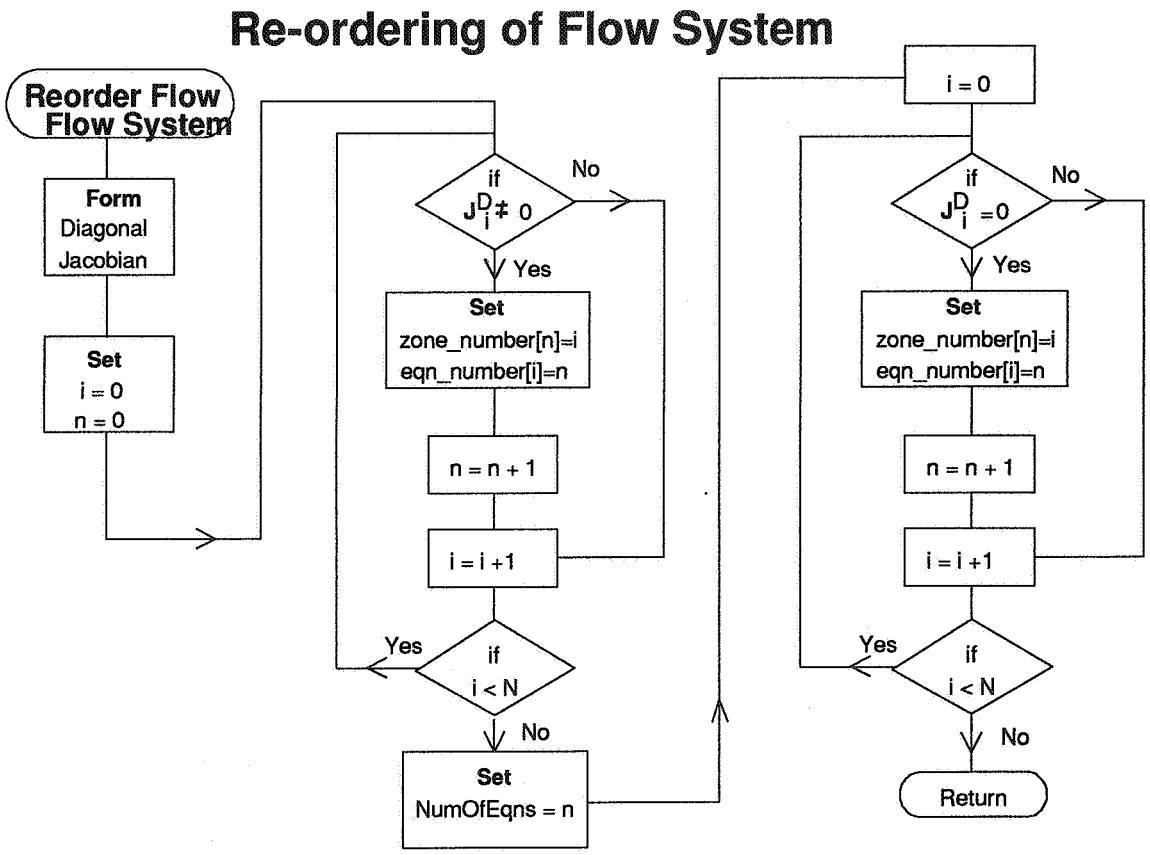


Figure 6. Schematic of Method for Re-ordering Flow System of Equations

The solution of the dynamic response of the transient contaminant dispersal system of equations uses the predictor-corrector model explained in the reports of Axley[4,5]. Three vector work arrays, **dE**, **dC** and **ddC** are allocated along with the storage for the modified dispersal matrix $\mathbf{K}_M$. The initial rate of change of the contaminant concentrations is estimated from the dispersal equations and used to estimate the time step dt to guarantee at least an accuracy of 5 percent in the solution. The time is incremented and the modified dispersal matrix is calculated. For each time step dt, an initial estimate of the increment **dC** is determined and then a correction is calculated and the concentration is updated. If the print interval is reached, the concentrations of the massless nodes are calculated and the concentrations for each zone are printed both to the screen and the output file. This process continues till the end of the time interval is reached.

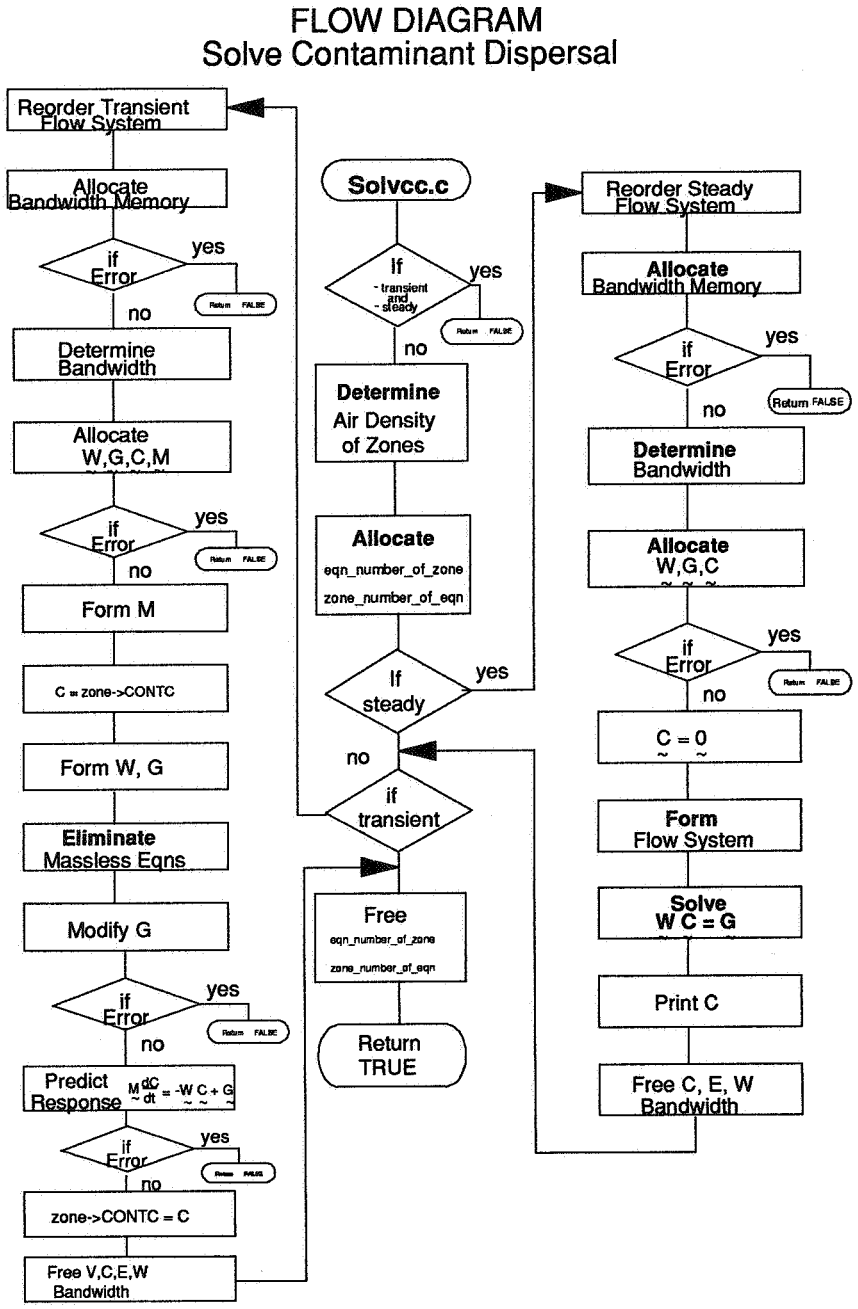


Figure 7. Schematic of Solution Method for Contaminant Dispersal Analysis Used by **CONTAM88.**

If the number of contaminants is greater then zero, **CONTAM88** checks to see if either a transient or steady state analysis of the contaminant dispersal problem is desired. **CONTAM88** can do either one or both for each set of weather conditions. If none is wanted, it returns to the main program, otherwise it determines the air density of each zone and allocates memory for two integer arrays eqn_number_of_zone which will later give the contaminant equation number of the

zone and zone_number_of_eqn which will give the zone of the corresponding to the contaminant equation. **CONTAM88** will reorder the contaminant dispersal equations to eliminate isolated zones and for the transient analysis eliminate zones with zero volume. If a steady state analysis is desired, **CONTAM88** formulates and solves this problem. It first reorders the flow system such that zones with non-zero airflows on the diagonal of the dispersal matrix appear as the first $N_f \cdot M$ unknown contaminant concentrations and the remaining $(N-N_f) \cdot M$ are contaminant concentrations for isolated zones where N is the number of zones in the building and M is the number of contaminants. The number N_f is the number of non-isolated zones and is determined by the reordering algorithm.

The bandwidth of the dispersal matrix is then determined and the memory storage required for the dispersal matrix W, the contaminant generation vector E and the unknown contaminant concentrations C is allocated. If an error occurs in the allocation of memory, the solution is aborted. The contaminant dispersal system is then formulated and solved. The results are then written to both a file and the screen. The memory required to solve the steady state dispersal problem is then freed. **CONTAM88** then checks to see if a transient contaminant dispersal solution is wanted. If so, it reorders the dispersal system zones which both are isolated and have zero volume are eliminated from the solution. It then partitions the remaining equations so the non-zero volume zones are first and the remaining zones are those with zero volume. The bandwidth of the renumbered dispersal system is then determined, memory allocated for the dispersal matrix **W**, the contaminant generation vector **E**, the reordered zone mass vector **V** and the transient concentrations **C** of the reordered system. If an error occurs in the allocation of memory, the solution is aborted. The dispersal mass vector **V** is then formed, the initial concentrations are retrieved from the zone data structures and the dispersal matrix **W** and the generation vector **E** are formed. If there are equations with zero volumes (massless equations), they are eliminated from the transient system of dispersal equations [1]. The dynamic response of the system

$$\overleftrightarrow{\mathbf{M}}\frac{d\vec{\mathbf{C}}}{dt} = \overleftrightarrow{\mathbf{W}}\vec{\mathbf{C}} + \vec{\mathbf{E}}$$

is solved. Figure 3 gives the schematic of the logic for solving the transient system of equations. After the transient solution is finished for the time interval for which the weather conditions are valid, the results of are storage in the zone data structures and the memory required for the transient solution is freed. **CONTAM88** then frees the memory for the arrays eqn_number_of_eqn and zone_number_of_eqn and returns to the main routine.

---

1 The massless equations are in essence a steady state subsystem which needs only to be solved when the results of the transient solution are written to either the screen or a file.

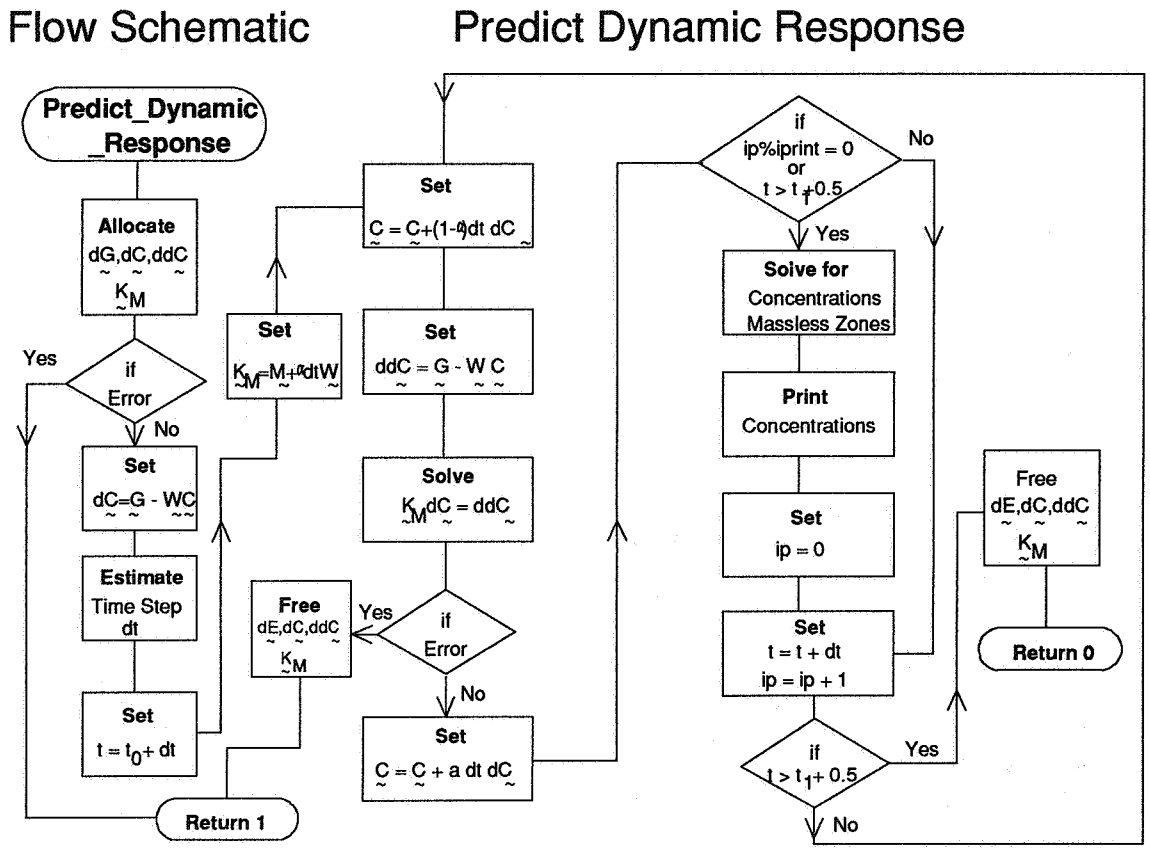


Figure 8. Schematic of Solution of Transient Contaminant Dispersal Equations

4 Theory of Airflow and Contaminant Dispersal Analysis Used by CONTAM88

The basic theory of airflow analysis used by **CONTAM88** is based on the idealization that a building can be treated as a series of pressure nodes connected by a group of flow paths and that the flow through each flow path is a function of the pressures at the nodes connected by the flow paths. In buildings, the pressure nodes are usually called zones and the flow paths will be called flow elements. A zone requires three parameters to specify its state for flow analysis, its pressure, its temperature and its volume. In **CONTAM88** the pressure is considered an unknown to be solved for by the system of equations formulated by **CONTAM88** and the zone's temperature and volume are considered specified, though not necessarily constant. A flow element is specified by giving the numbers of the pressure nodes it connects and the functional form of the relation between the flow through the element and the element node pressures.

CONTAM88 used two types of flow elements: openings and fans. Each of these elements are two node elements. If we consider the index i, i = 1,2 than the element can be defined by determining the quantities (${}^{e}f_i, {}^{e}p_i, {}^{e}T_i$) where ${}^{e}f_i$ is the flow into i^{th} node from the element and ${}^{e}p_i$ and ${}^{e}T_i$ are the nodes pressure and temperature respectively. All of these quantities are not independent. Conservation of mass on the element requirements requires that

$$ {}^{e}f_1 = -{}^{e}f_2 \equiv {}^{e}f(\delta p^{e}) \qquad (1) $$

where $\delta p^{e} = {}^{e}p_2 - {}^{e}p_1 + {}^{e}p_w + {}^{e}p_s$

and $ {}^{e}p_s = g(\rho_2 h_2 - \rho_1 h_1)$,stack pressure

and $ {}^{e}p_s = \frac{1}{2} pc[m_s]\rho_0 V(H_s)^2 (\delta_{0n_2} - \delta_{0n_1})$,wind pressure

In the above ${}^{e}p_w$ is the wind pressure on the element, ${}^{e}p_s$ is the stack pressure across the element. h_1 and h_2 are the heights of the element above the reference height of the zones n_1 and n_2, where n_1 is the zone number of the zone from which the flow comes (if ${}^{e}f$ is positive) and n_2 is the zone number of the zone to which the flow goes. ρ_1 and ρ_2 are the density of air in the zone n_1 and n_2 respectively. A convention that **CONTAM88** uses is that the number of the exterior zone is 0. In the expression for the wind pressure p_w, $V(H_s)$ is the wind speed at the height of the surface containing the opening.

$$ V(H_s) = V_m \left(\frac{H_s}{10} \right)^{0.143} \qquad (2) $$

where V_m is the measured meteorological wind speed at a height of 10 meters.

$pc[m_s]$ is the m_s^{th} pressure coefficient of the surface. The integer m_s is determined from the wind direction a_w and the azimuth angle a_z of the surface which contains the opening by determining the sector relative to the surface from which the wind comes:

$$ m_s = n_s \cdot \frac{\phi}{360} + \frac{1}{2} \qquad (4) $$

where $\phi = a_w - a_z$ if $a_w \geq a_z$

$= 360 - (a_z - a_w)$ if $a_w < a_d$

if $m_s > n_s$ then $m_s = 1$;

In the above expression, n_s is the number of surface coefficients of the surface containing the opening.

Flow-Pressure Equation for an Opening.

In **CONTAM88** the mass flow air through the opening is related to the pressure drop across the opening by a power-law equation in the turbulent flow regime and by a quadratic equation in the laminar flow regime.

$$^{e}f = \sqrt{\rho_i} C \delta p |\delta p|^{x-1} \quad \text{if} \quad |\delta p| \geq \Delta p_t \tag{5}$$

$$= \sqrt{\rho_i}\, \delta p (\alpha + \beta |\delta p|) \quad \text{if} \quad |\delta p| < \Delta p_t$$

where

$$\alpha = (2-x) C \Delta p_t^{x-1}$$

$$\beta = (n-1) C \Delta p_t^{x-2}$$

$$\Delta p_t = \frac{0.0034}{C} \tag{6}$$

and ρ_i is the inlet air density: $i = 2$ if $\delta p \geq 0$,$i = 1$ if $\delta p < 0$

In the above equation Δp_t is the transient pressure at which the flow through the element changes from laminar to turbulent. Though this form of a relationship between the flow is physically correct, its real purpose in the theory is to assure convergence of the mathematical method used to solve the flow system of equations. Note that if a linear region is not assumed about 0, the derivative of the flow equation with respect to pressure is infinite. In contrast to AIRNET[9], **CONTAM88** requires that both the f and its derivative are continuous at the transient pressure. The expression for α and β are derived from these conditions.

One can justify the expression for the transition pressure by the following reasoning: Let U be the velocity of air passing through the opening.

$$U = \frac{f}{\rho A_l} \tag{7}$$

$$A_l = \frac{C}{k} \tag{8}$$

$$k \equiv \frac{\delta P_{ref}^{0.5-x}}{\sqrt{2}} \tag{9}$$

where A_l is the leakage area of the opening and k is the discharge coefficient. The reference pressure δP_{ref} is usually chosen to be 4 Pa in the United States. Then the Reynolds number of the flow through the opening can be expressed as

$$Re \equiv U \frac{L}{\nu} \tag{10}$$

ν is the viscosity of air $= 1.5 \cdot 10^{-5}$

L is a representative length $\equiv \sqrt{A_l}$

In the turbulent flow region, the Reynolds number can be related to the pressure drop across the flow path by

$$Re = \sqrt{\frac{2A_l}{\rho}\frac{k}{\nu}}\delta p^x \tag{11}$$

The transition pressure Δp_t can thus be estimated by the expression

$$\Delta p_t = \left(\sqrt{\frac{\rho}{2A_l}\frac{Re_t}{k}}\nu\right)^{\frac{1}{x}} \tag{12}$$

$$= \left(\sqrt{\frac{\rho}{2Ck}}Re_t\nu\right)^{\frac{1}{x}}$$

$$\cong \frac{\rho}{2Ck}(Re_t\nu)^2$$

where Re_t is the Reynolds number at which the flow becomes turbulent.

If one assumes that the transition to turbulence occurs for a Reynolds number of about 5000, one obtains the previous given expression for Δp_t.

Fan Flow Element

The fan flow element used by **CONTAM88** is a simple fixed flow element whose flow pressure relation is given by

$$f(\delta p^e) = F_0 \tag{13}$$

If F_0 is positive , flow is from node 2 to node 1 of the element; if F_0 is negative, the flow is from node 1 to node 2 of the element. There is no mathematical difficulty in introducing fan curves in the theory; however the present version of **CONTAM88** treats fans as constant flow elements.

Formation of the Flow System

The system of flow equations which are to be solved for the pressures in each zone can be formulated by defining the flow and pressure vectors for each element as

$$\vec{\mathbf{p}}_e \equiv \begin{bmatrix} {}^e p_1 \\ {}^e p_2 \end{bmatrix} \tag{14}$$

$$\vec{\mathbf{f}}_e \equiv \begin{bmatrix} {}^e f_1 \\ {}^e f_2 \end{bmatrix} \tag{15}$$

If one defines the system pressure vector $\vec{\mathbf{p}}$ and zone flow vector $\vec{\mathbf{f}}$ such that

$$\vec{\mathbf{p}} \equiv \begin{bmatrix} p_1 \\ p_2 \\ \cdots \\ p_N \end{bmatrix} \quad \vec{\mathbf{f}} \equiv \begin{bmatrix} f_1 \\ f_2 \\ \cdots \\ f_N \end{bmatrix} \tag{16}$$

where p_i is the pressure of the i^{th} zone and f_i is the system of the mass flows into the i^{th} zone minus the sum of the flows out of the i^{th} zone from all elements and N is the number of independent pressures nodes for the system.

One can then consider the mapping of the element node pressures into the zone pressures given by the equation:

$$\vec{\mathbf{p}}_e = \overleftrightarrow{\mathbf{B}}_e \vec{\mathbf{p}} \tag{17}$$

where

$$\overleftrightarrow{\mathbf{B}}_e \equiv [B_{ij}^e] = \begin{bmatrix} B_{11} & B_{12} & \cdots & B_{1N} \\ B_{21} & B_{22} & \cdots & B_{2N} \end{bmatrix} \tag{18}$$

where $B_{ij}^e = \delta_{il}\delta_{jn_1} + \delta_{i2}\delta_{jn_2}$

In equation (18), n_1 is the zone number of element node 1 and n_2 is the zone number of element node 2. The zone numbers n_1 and n_2 are determined by the building description processor **NBSAVIS** and stored by **CONTAM88** in the data structure for each element.

The system flow vector $\vec{\mathbf{f}}$ can then be formed from the element flow vectors by the operation:

$$\vec{\mathbf{f}}(\vec{\mathbf{p}}) = \sum_{\forall e} \overleftrightarrow{\mathbf{B}}_e^{\,T} \vec{\mathbf{f}}_e(\overleftrightarrow{\mathbf{B}}_e \vec{\mathbf{p}}) \tag{18}$$

where the summation is over all flow elements.

The conservation of mass at each zone requires that

$$\vec{\mathbf{f}}(\vec{\mathbf{p}}) = \frac{d\vec{\mathbf{m}}}{dt} \tag{19}$$

where $\vec{\mathbf{m}}$ is the vector of zone masses.

$$\vec{\mathbf{m}} \equiv \begin{bmatrix} \rho_1 V_1 \\ \rho_2 V_2 \\ \cdots \\ \rho_N V_N \end{bmatrix} \tag{20}$$

where ρ_i is the air density of the i^{th} zone and V_i is the zone volume.

The above set of equations constitutes as nonlinear system of N equations for the N zone pressures.

In most cases, the right hand side of the flow equations is zero (that is, there is no rate of change of mass inside the zone). However there instances in which these terms cannot be neglected. For example, in the case of change zone temperature or varying barometric pressure, the zone air densities will change with time. If one is modelling two sections of an elevator shaft with a moving elevator, the volumes of the two zones will change with time. **CONTAM88** does however consider the right hand side of the flow equations to be negectible.

Solution Technique for Solving the Flow System

The system of flow equations is solved using a Newton-Raphson technique. If one defines the Jacobian matrix

$$\overleftrightarrow{\mathbf{J}}(\vec{\mathbf{p}}) \equiv -\frac{\partial \vec{\mathbf{f}}}{\partial \vec{\mathbf{p}}} \tag{21}$$

then the flow system can be solved iteratively by the procedure

$$\overleftrightarrow{\mathbf{J}}(\vec{0})\Delta\vec{\mathbf{p}}^{0} = \vec{\mathbf{f}}(\vec{0}) - \frac{d\vec{\mathbf{m}}}{dt} \tag{22}$$

$${}^{1}\vec{\mathbf{p}} = \Delta\vec{\mathbf{p}}^{0}$$

then for $\alpha = 1\ldots$

$$\overleftrightarrow{\mathbf{J}}({}^{\alpha}\vec{\mathbf{p}})\Delta\vec{\mathbf{p}}^{\alpha} = \vec{\mathbf{f}}({}^{\alpha}\vec{\mathbf{p}}) - \frac{d\vec{\mathbf{m}}}{dt} \tag{23}$$

$${}^{\alpha+1}\vec{\mathbf{p}} = {}^{\alpha}\vec{\mathbf{p}} + \overleftrightarrow{\omega}\Delta\vec{\mathbf{p}}^{\alpha}$$

until the convergence criteria satisfied or $\alpha > \max$

In the above scheme, ω_i are relaxation factors used to prevent oscillations in the convergence of the solution.

It can be shown that the Jacobian matrix of the flow system can be formed from the Jacobian matrices of the flow elements:

$$\overleftrightarrow{\mathbf{J}}(\vec{\mathbf{p}}) = \sum_{\forall e} \overleftrightarrow{\mathbf{B}}_e^{T} \overleftrightarrow{\mathbf{J}}_e \overleftrightarrow{\mathbf{B}}_e \tag{24}$$

where $\overleftrightarrow{\mathbf{J}}_e \equiv -\frac{\partial \vec{\mathbf{f}}_e}{\partial \vec{\mathbf{p}}_e}$

For two node flow elements in which the flow depends on the difference of the nodal pressures if can be shown that the element Jacobian has the form:

$$\vec{\mathbf{J}}_e = d^e f \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \quad \text{if } n_1 \neq 0 \text{ and } n_2 \neq 0 \tag{25}$$

$$= d^e f \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \quad \text{if } n_2 = 0$$

$$= d^e f \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \quad \text{if } n_1 = 0$$

where $d^e f \equiv \dfrac{d^e f}{d\delta p}$

Note that since the element Jacobian matrices are symmetric, the flow system Jacobian is symmetric. Also, since $d^e f > 0$, the diagonal elements are positive and the off-diagonal elements are negative or zero and the sum of the absolute value of off-diagonal elements is equal to or less than the diagonal elements for each element matrix, the same holds true for the system Jacobian. Therefore, the system Jacobian is a symmetric M-matrix and if the a diagonal element of the system flow matrix is zero, then all elements in the row and column which contain the diagonal element are zero. That the Jacobian is a M-matrix has two important consequences for the theory: First, the solution of the system of equations (23) for $\Delta \mathbf{p}$ can be accomplished by LU decomposition without pivoting. That pivoting is not required allows the use of compact and skyline matrix storage techniques which could be destroyed by pivoting. Second, that a zone is isolated can be determined by checking if the diagonal element of the Jacobian is zero. Note also from the form of the element pressure-flow relation, a zero diagonal element at any pressure implies that the flow coefficients of all elements connected to the zone are zero and therefore diagonal element is identically zero ate all pressures.

Convergence Criteria

CONTAM88 uses a combination of an absolute and relative convergence criteria on the components of the zone flow vector.

if $|f_i| > \varepsilon_1$ and $|f_i| / \sum_{\forall e} |^e f_i| > \varepsilon_2$ for any i, then cnvg = false otherwise convg = true

The default values for ε_1 and ε_2 are 0.000001 and 0.0001 respectively.

Contaminant Dispersal

The contaminant dispersal theory implemented by **CONTAM88** is a slightly modified version of the well-mixed zone theory presented by Axley[4,5]. The theory of **CONTAM87** has been expended to include the phenomena of plateout without resorting to the use of dummy kinetic reaction elements. The convection diffusion element of **CONTAM87** has not been included in **CONTAM88**. The contaminant dispersal theory of **CONTAM88** is a multi-contaminant theory which includes the physical effects of the transport of contaminants by air movement, the removal of contaminants by plateout and the kinetic reaction of contaminants within a zone. The theory also includes contaminants sources which can depend on the exterior-interior pressure differences across the building (for example, pressure driven radon sources). The basic equations for the contaminant dispersal in a building can be derived from the conservation of mass for contaminant specie in each zone of the building:

$$\frac{d}{dt}(m_i C^{\alpha}{}_i) = -(f_i^{(out)} + R_i^{\alpha})C^{\alpha}{}_i + \sum_{\forall e}\sum_{j=0}^{N}(1-{}^{e}\eta^{\alpha})\,{}^{e}f_{ij}^{(in)}C^{\alpha}{}_j + m_i\sum_{\beta=1}^{M}\kappa^{i}_{\alpha\beta}C^{\beta}{}_i + g^{\alpha}{}_i \quad , \text{i=1...N} \tag{26}$$

where $m_i \equiv \rho_i V_i$

In equation 26, $C^{\alpha}{}_i$ is the concentration of the α^{th} contaminant specie in the i^{th} zone, m_i is the mass of air in the i^{th} zone, $f^{(out)}{}_i$ is the sum of all air flows out of the i^{th} zone, $f^{(in)}{}_{ij}$ is the sum of all air flows into the i^{th} zone for the j^{th} zone, $R^{\alpha}{}_i$ is the removal rate by plateout or deposition of the α^{th} specie in the i^{th} zone, ${}^{e}\eta^{\alpha}$ is the filter efficiency for the e^{th} flow element and the α^{th} specie, $\kappa^{i}_{\alpha\beta}$ is the kinetic reaction coefficient in the i^{th} zone between α^{th} and β^{th} species and $g^{\alpha}{}_i$ is the generation rate of the α^{th} specie in the i^{th} zone. In equation 26, j = 0 denotes the exterior zone; that is $f^{(in)}{}_{i0}$ is the flow into the i^{th} zone form the exterior and ${}^{\alpha}C_0$ is the concentration of the α^{th} specie in the exterior zone. **CONTAM88** uses mass flow rates for air flows and mass concentration units per unit mass of air for the concentration of pollutant species in its contaminant dispersal calculations. Note also the definition of the kinetic rate matrix is such that a positive coefficient produces an increase in contaminant mass (**CONTAM87** uses the oppositive sign convention).

If for each flow element we define ${}^{e}f^{(out)}{}_i$ as the flow out of the zone through the flow element and ${}^{e}f^{(in)}{}_{ij}$ as the flow into the i^{th} zone from the j^{th} zone then

$${}^{e}f_i^{(out)} = \delta_{in_2}\,{}^{e}f \quad \text{if} \quad {}^{e}f \geq 0 \tag{27}$$

$$= -\delta_{in_1}\,{}^{e}f \quad \text{if} \quad {}^{e}f < 0$$

$${}^{e}f_{ij}^{(in)} = \delta_{in_1}\delta_{jn_2}\,{}^{e}f \quad \text{if} \quad {}^{e}f \geq 0 \tag{28}$$

$$= -\delta_{in_2}\delta_{jn_1}\,{}^{e}f \quad \text{if} \quad {}^{e}f < 0$$

then $f^{(out)}{}_i$ and $f^{(in)}{}_i$, the total flows out of and into the ith zone respectively, are given by:

$$f_i^{(out)} = \sum_{\forall e}{}^{e}f_i^{(out)} \tag{29}$$

$$f_i^{(in)} = \sum_{j=0}^{N}\sum_{\forall e}{}^{e}f_{ij}^{(in)} \tag{30}$$

From the conservation of mass for the air in the i^{th} zone (eqn 19), it can be shown that $f^{(out)}{}_i$ and $f^{(in)}{}_i$ are related by

$$\frac{dm_i}{dt} = -f_i^{(out)} + f_i^{(in)} \tag{31}$$

Therefore, equation 26 can be written as

$$m_i\frac{d^{\alpha}C_i}{dt} = -{}^{\alpha}R_i\,{}^{\alpha}C_i - \sum_{j=0}^{N}\sum_{\forall e}\{{}^{e}f_{ij}^{(in)\alpha}C_i - (1-{}^{e}\eta^{\alpha})\,{}^{e}f_{ij}^{(in)\alpha}C_j\} + m_i\sum_{\beta=1}^{M}\kappa^{i}_{\alpha\beta}\,{}^{\beta}C_i + g_i^{\alpha} \tag{32}$$

Note that equation (32) is the correct form of the conservation of species equation for the case where m_i are not constant but time varying functions (the coefficient of ${}^{\alpha}C_i$ in the equation for ${}^{\alpha}C_i$ is the sum of the flows into the i^{th} zone, not necessarily the flow out of the zone).

If one defines for each zone, the specie's vector $\vec{C}_i$

$$\text{where} \quad \vec{C}_i = \begin{bmatrix} {}^1C_i \\ {}^2C_i \\ \cdots \\ {}^MC_i \end{bmatrix} \tag{33}$$

and for each element the specie element vector ${}^e\vec{C}^{\alpha}$

$$\text{where} \quad {}^e\vec{C}^{\alpha} \equiv \begin{bmatrix} {}^{\alpha}C_1^e \\ {}^{\alpha}C_2^e \end{bmatrix} \tag{34}$$

and forms the system contaminant vector $\vec{C}$

$$\vec{C} \equiv \vec{C}_1 \oplus \vec{C}_2 \oplus \ldots \oplus \vec{C}_N = \begin{bmatrix} \vec{C}_1 \\ \vec{C}_2 \\ \cdots \\ \vec{C}_N \end{bmatrix} \tag{35}$$

the contaminant dispersal equations can be written in the form

$$\vec{\vec{m}} \frac{d\vec{C}}{dt} = -\vec{\vec{W}} \cdot \vec{C} + \vec{g}' \tag{36}$$

where $\vec{\vec{m}}$ is the system mass matrix, $\vec{\vec{W}}$ is the total system dispersal matrix and $\vec{g}'$ is the total system contaminant generation vector.

The system mass matrix is

$$\vec{\vec{m}} \equiv \vec{\vec{m}}_1 \oplus \vec{\vec{m}}_2 \ldots \oplus \vec{\vec{m}}_N = \begin{bmatrix} \vec{\vec{m}}_1 & \vec{\vec{0}} & \vec{\vec{0}} & \cdots & \vec{\vec{0}} \\ \vec{\vec{0}} & \vec{\vec{m}}_2 & \vec{\vec{0}} & \cdots & \vec{\vec{0}} \\ \vec{\vec{0}} & \vec{\vec{0}} & \cdots & \cdots & \vec{\vec{0}} \\ \vec{\vec{0}} & \vec{\vec{0}} & \cdots & \vec{\vec{0}} & \vec{\vec{m}}_N \end{bmatrix} \tag{37}$$

where $\vec{\vec{m}}_i$ is the mass matrix of the i^{th} zone

$$\vec{\vec{m}}_i \equiv m_i \vec{\vec{I}}_M \tag{38}$$

where $\vec{\vec{I}}_M$ is the $M \times M$ unit matrix

and the system dispersal matrix $\overleftrightarrow{\mathbf{W}}$ is given by

$$\overleftrightarrow{\mathbf{W}} \equiv \overleftrightarrow{\mathbf{R}} + \overleftrightarrow{\mathbf{F}} - \overleftrightarrow{\mathbf{K}} \tag{39}$$

where $\overleftrightarrow{\mathbf{R}}$ is the removal rate matrix given by

$$\overleftrightarrow{\mathbf{R}} \equiv \overleftrightarrow{\mathbf{R}}_1 \oplus \overleftrightarrow{\mathbf{R}}_2 \oplus \ldots \oplus \overleftrightarrow{\mathbf{R}}_N = \begin{bmatrix} \overleftrightarrow{\mathbf{R}}_1 & \overleftrightarrow{\mathbf{0}} & \overleftrightarrow{\mathbf{0}} & \cdots & \overleftrightarrow{\mathbf{0}} \\ \overleftrightarrow{\mathbf{0}} & \overleftrightarrow{\mathbf{R}}_2 & \overleftrightarrow{\mathbf{0}} & \cdots & \overleftrightarrow{\mathbf{0}} \\ \overleftrightarrow{\mathbf{0}} & \overleftrightarrow{\mathbf{0}} & \cdots & \cdots & \overleftrightarrow{\mathbf{0}} \\ \overleftrightarrow{\mathbf{0}} & \overleftrightarrow{\mathbf{0}} & \cdots & \overleftrightarrow{\mathbf{0}} & \overleftrightarrow{\mathbf{R}}_N \end{bmatrix} \tag{40}$$

with

$$\overleftrightarrow{\mathbf{R}}_i \equiv \begin{bmatrix} {}^1R_i & 0 & 0 & \cdots & 0 \\ 0 & {}^2R_i & 0 & \cdots & 0 \\ 0 & 0 & \cdots & \cdots & \\ 0 & 0 & \cdots & \cdots & {}^MR_i \end{bmatrix} \tag{41}$$

and $\overleftrightarrow{\mathbf{F}}$ is the flow matrix

$$\overleftrightarrow{\mathbf{F}} \equiv \sum_{\forall e} \sum_{\alpha=1}^{M} {}^{\alpha}\overleftrightarrow{\mathbf{B}}^{eT} \cdot {}^{e}\overleftrightarrow{\mathbf{f}}^{\alpha} \cdot {}^{\alpha}\overleftrightarrow{\mathbf{B}}^{e} \tag{42}$$

where ${}^{e}\overleftrightarrow{\mathbf{B}}^{\alpha}$ is the mapping of the element species onto the system contaminant species

$${}^{e}\overleftrightarrow{\mathbf{B}}^{\alpha} \equiv [{}^{e}B^{\alpha}_{ij}] = \begin{bmatrix} {}^{e}B^{\alpha}_{11} & {}^{e}B^{\alpha}_{12} & \cdots & {}^{e}B^{\alpha}_{1N_c} \\ {}^{e}B^{\alpha}_{21} & {}^{e}B^{\alpha}_{22} & \cdots & {}^{e}B^{\alpha}_{2N_c} \end{bmatrix} \tag{43}$$

where ${}^{e}B^{\alpha}_{ij} = \delta_{in_{1\alpha}} + \delta_{jn_{2\alpha}} \quad i,j = 1 \ldots N_c$

and $n_{\alpha l} = M(n_l - 1) + \alpha \quad \text{if} \quad n_l \neq 0$

$= 0 \quad \text{if} \quad n_l = 0 \quad \text{for l=1,2}$

and ${}^{e}\overleftrightarrow{\mathbf{f}}^{\alpha}$ is the element flow matrix for α^{th} specie

$${}^{e}\overleftrightarrow{\mathbf{f}}^{\alpha} = {}^{e}f \begin{bmatrix} 1 & 0 \\ -(1-{}^{e}\eta^{\alpha}) & 0 \end{bmatrix} \quad \text{if} \quad {}^{e}f \geq 0 \tag{44}$$

$$= -{}^{e}f \begin{bmatrix} 0 & -(1-{}^{e}\eta^{\alpha}) \\ 0 & 1 \end{bmatrix} \quad \text{if} \quad {}^{e}f < 0$$

and $\vec{\mathbf{g}}'$ is the system generation vector

$$\vec{\mathbf{g}}' \equiv \vec{\mathbf{g}}'_1 \oplus \vec{\mathbf{g}}'_2 \oplus \ldots \oplus \vec{\mathbf{g}}'_N = \begin{bmatrix} \vec{\mathbf{g}}'_1 \\ \vec{\mathbf{g}}'_2 \\ \cdots \\ \vec{\mathbf{g}}'_N \end{bmatrix} \tag{45}$$

where $\vec{\mathbf{g}}'_i \equiv \begin{bmatrix} {}^1g'_i \\ {}^2g'_i \\ \cdots \\ {}^Mg'_i \end{bmatrix}$

with ${}^\alpha g'_i = g^\alpha{}_i + \sum_{\forall e} {}^e f_{i0}^{(\text{in})} (1 - {}^e\eta^\alpha)^\alpha C_0 \quad \text{for } \alpha = 1 \ldots M$

and $\overleftrightarrow{\mathbf{K}}$ is the system reaction matrix:

$$\overleftrightarrow{\mathbf{K}} \equiv \overleftrightarrow{\mathbf{K}}_1 \oplus \overleftrightarrow{\mathbf{K}}_2 \oplus \ldots \oplus \overleftrightarrow{\mathbf{K}}_N = \begin{bmatrix} \overleftrightarrow{\mathbf{K}}_1 & \overleftrightarrow{\mathbf{0}} & \overleftrightarrow{\mathbf{0}} & \cdots & \overleftrightarrow{\mathbf{0}} \\ \overleftrightarrow{\mathbf{0}} & \overleftrightarrow{\mathbf{K}}_2 & \overleftrightarrow{\mathbf{0}} & \cdots & \overleftrightarrow{\mathbf{0}} \\ \overleftrightarrow{\mathbf{0}} & \overleftrightarrow{\mathbf{0}} & \cdots & \cdots & \overleftrightarrow{\mathbf{0}} \\ \overleftrightarrow{\mathbf{0}} & \overleftrightarrow{\mathbf{0}} & \cdots & \overleftrightarrow{\mathbf{0}} & \overleftrightarrow{\mathbf{K}}_N \end{bmatrix} \tag{46}$$

where $\overleftrightarrow{\mathbf{K}}_i = [\kappa^i_{\alpha\beta}]$

$$= \begin{bmatrix} \kappa^i_{11} & \kappa^i_{12} & \cdots & \kappa^i_{1M} \\ \kappa^i_{21} & \kappa^i_{22} & \cdots & \kappa^i_{2M} \\ \cdots & \cdots & \cdots & \cdots \\ \kappa^i_{M1} & \kappa^i_{M2} & \cdots & \kappa^i_{MM} \end{bmatrix}$$

The method used in **CONTAM88** for forming and solving the system of equations for the building contaminant dispersal are discussed in the next sections.

Data Structure

CONTAM88 uses a link list data structure for storing the data for the zones, openings, fans, kinetic reaction data and surface data. These link lists are allocated dynamically when the corresponding data is read. The contaminant physical data is stored in a series of dynamically allocated arrays.

The zone data structure is shown in figure 9. The zone data is stored in the structures of a linked list. The zone data structure contains physical data which describes the zone (its volume, temperature, height above the reference), variables which are the results of the calculations (zone air density, total mass of air in the zone, the zone pressure and a pointer to the results of the last transient contaminant dispersal analysis) and pointers to arrays which describe the contaminant sources (generations rates, removal rates, coefficients and exponents of pressure driven sources) in the zone and the kinetic reaction occurring in the zone. Each zone data structure contains a

point to the data structure of the next zone. The pointer, FirstZone, to the head of the list is a global variable. The last data structure is the list is contains a NULL pointer. In **CONTAM88**, the first zone data structure (pointed to by FirstZone) contains the outside ambient conditions. The first building zone is the second zone data structure (pointed to by FirstZone->NextZonePtr). The data contained in the file PROJECT.ZON are stored in the link list of zone data structures.

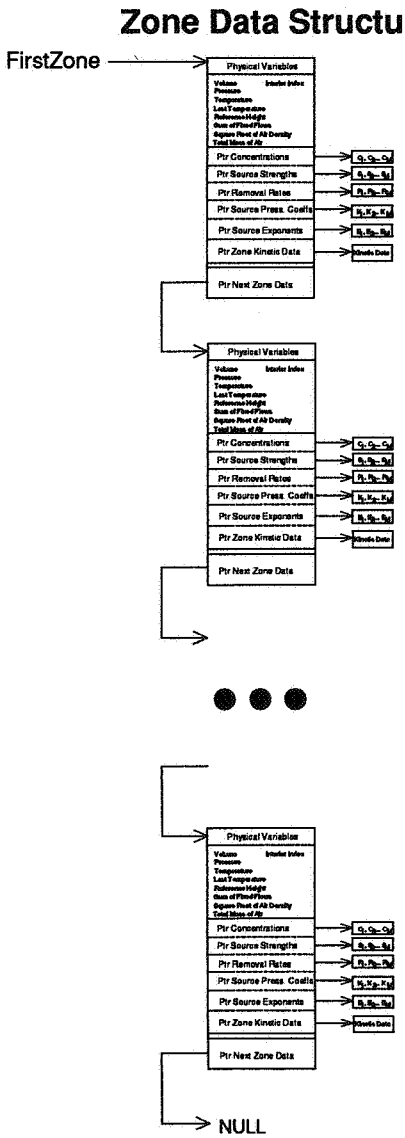
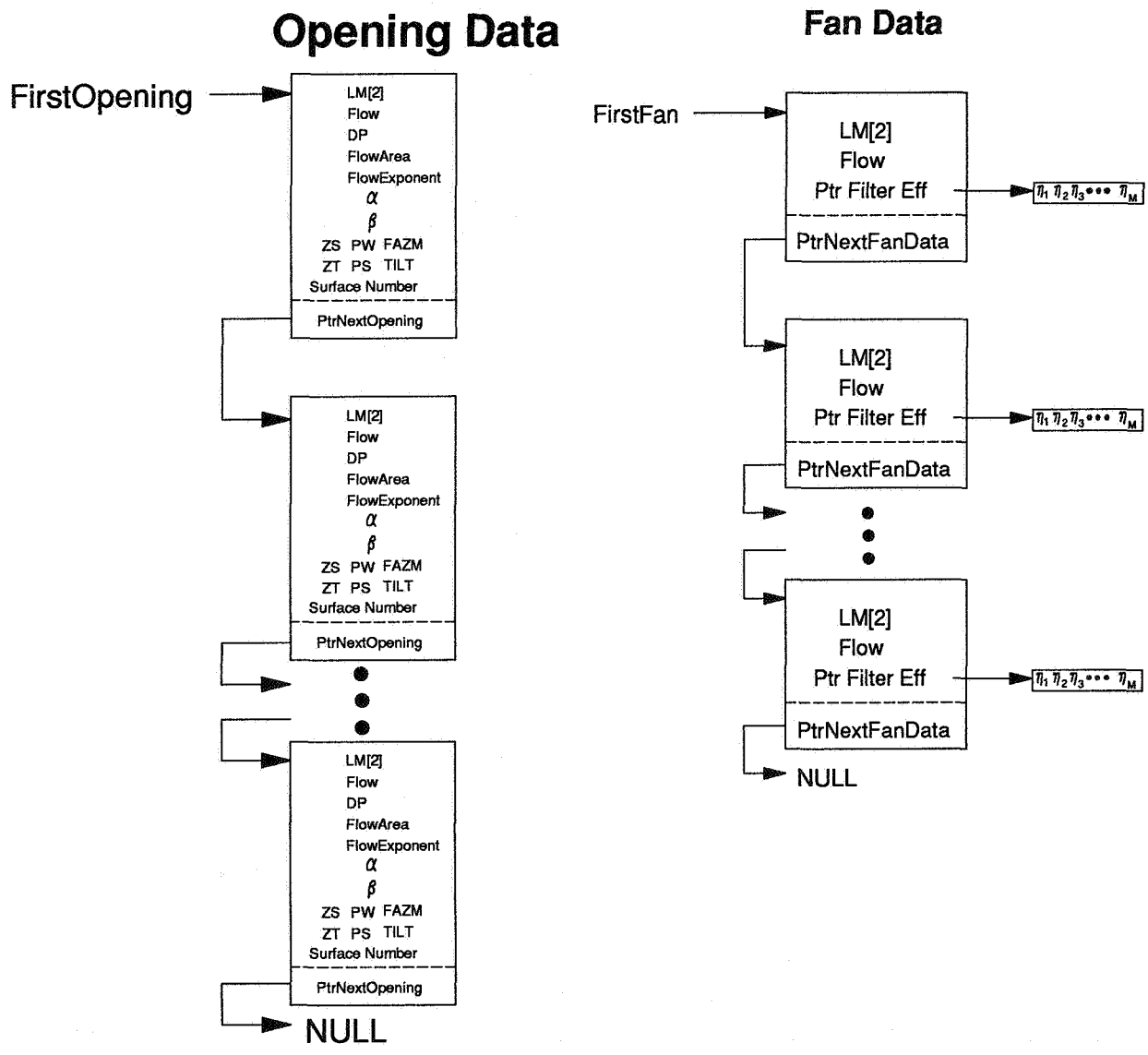


Figure 9. Zone Data Structure

The opening and fan data structures are shown in figure 9. The data for openings and fans are also stored in linked lists. The data structure of an opening contains the physical description of the opening contained in the data file PROJECT.OPN, the zones connected by the opening ,(LM[2] - where LM[0] = n_1, LM[1] = n_2 ,the zones numbers of the first and second node of the element), the flow coefficient, the flow exponent, the opening stack height, the height of the sur-

face containing the opening, the azimuth angle and tilt of the wall containing the opening and the number of the surface containing the opening. The opening data structure also contains results derived from these quantities: the transient pressure and the laminar region coefficients. During the airflow calculations the wind pressure, stack pressure, the flow through the opening and the pressure drop across the opening are storage in the zone data structure. Thus these quantities do not have to be recalculation very time they are needed. The fan data structure consist of the number of the zones connect by the fan (LM[2] has the same meaning as for an opening), the fan flow rate and a pointer to the efficiencies of any contaminant removal filters the fan may have. The head of each of these linked lists is determined by the global variables FirstOpening and FirstFan, respectively. Each list is NULL terminated.



Opening Data Structure Fan Data Structure

Figure 10. Data Structure of Flow Elements

The data structure for the kinetic reaction data and the surface data are shown in figure 10. Each of these data are stored in simple linked lists which contain only the physical data which describes the kinetic reactions and the surfaces pressure coefficients respectively. The kinetic data structures contains a pointer to a dynamically allocated matrix (see the following section on matrix storage techniques) of the reaction coefficients. The surface data structures contains an integer indicating the number of pressure coefficients of the surface (this can vary from surface to surface) and a pointer to a dynamically allocated array containing the pressure coefficients. The global variables which point the headers of these linked lists are FirstKineticData and First-Surface.

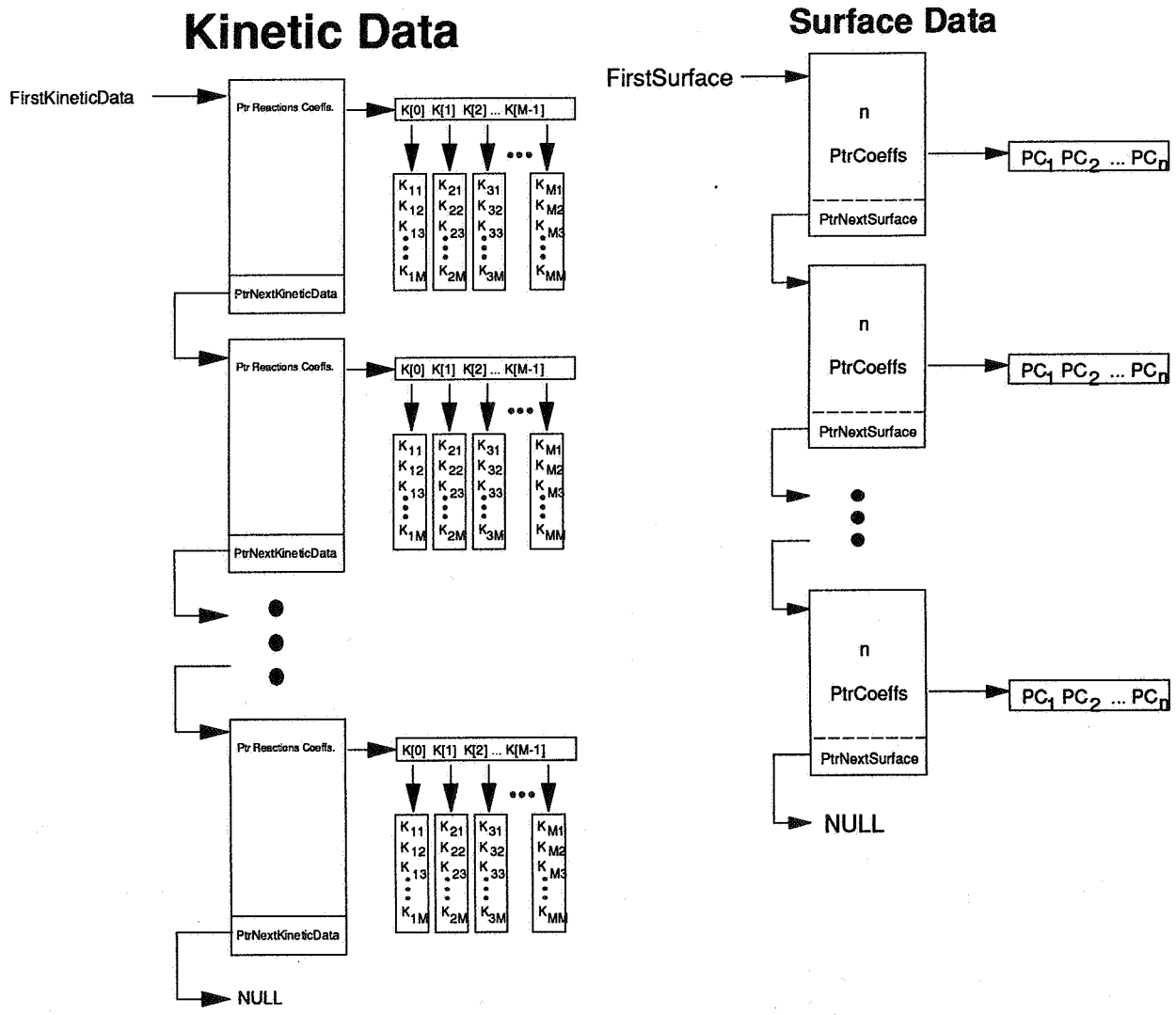


Figure 11. Kinetic Data Structure

Surface Data Structure

The data describing the properties of the contaminants are stored in dynamically allocated arrays: an array of string variable containing the contaminant name, an array of doubles containing the radioactivity of the contaminant, an array of doubles containing specific density of the contaminant, a boolean array indicating where the results are to be expressed in mass of air or volume of air and an array of integers indicating the units to be used in outputting the concentrations results for the contaminant. There is also an array of string variables which contain the name of the zones.

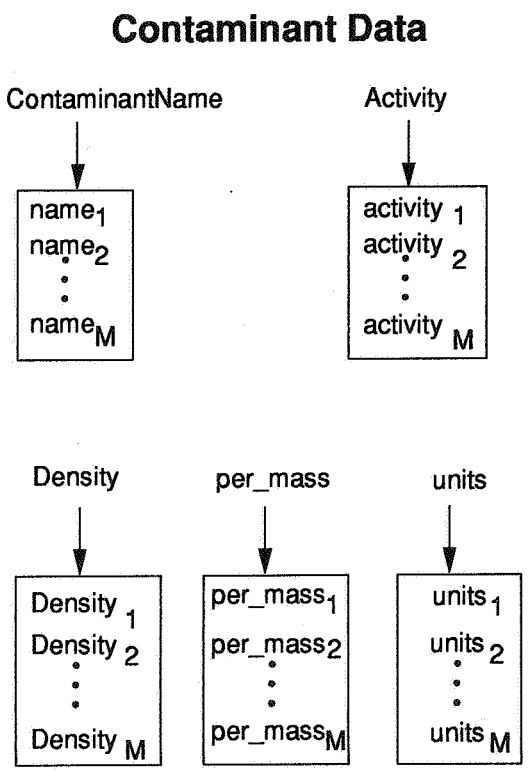


Figure 12. Contaminant Data Structure

The Use of CONTAM88 Data Structure to Form the Air Flow and Contaminant Dispersal Equations

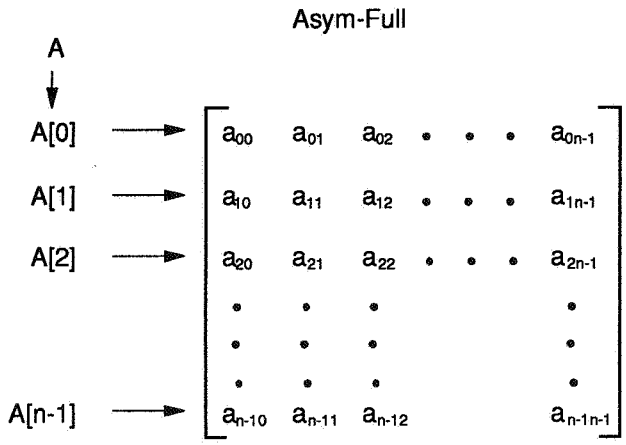
The above data structure permits a simple and compact method for forming the air flow and contaminant dispersal systems of equations. The equations describing to the forming of the system Jacobian $\overleftrightarrow{\mathbf{J}}$ (eqn 24) and the system contaminant dispersal matrix $\overleftrightarrow{\mathbf{W}}$ contains theoretical matrices $\overleftrightarrow{\mathbf{B}}_e$, ${}^e\overleftrightarrow{\mathbf{B}}^\alpha$ etc. These matrices are never really form in the assembly of the system equations. The lists of elements (for the air flow analysis, the opening and fan linked lists; for the contaminant dispersal matrix, the opening, fan, and zone link lists) are transversed, the element matrices are construction and added to the system matrix using the integer array which contains the equations number of nodes of the element. The following indicates this process for the air flow Jacobian

"C" Code Forming Air Flow Equations

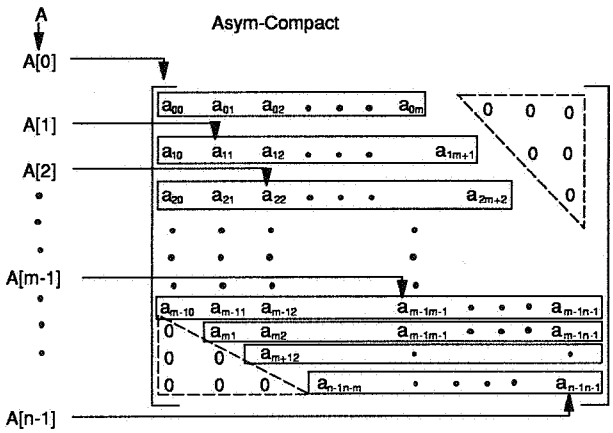
```
/* Form Flow Jacobian and Zone Flows */
  LastOpening = FirstOpening;
  i = 0;
  while(LastOpening != NULL)
  {
    i++;
    n = LastOpening->AFSPTR[0];
    m = LastOpening->AFSPTR[1];
    LastZone = ZonePointer(n);
    NearZone = ZonePointer(m);
/* Calculate pressure drop across opening */
    LastOpening->DP = ZonePointer(m)->PZ - ZonePointer(n)-
>PZ
                      +LastOpening->PS + LastOpening->PW;
/* Calculate flow through openig */
    LastOpening->Flow = 0.0;
    DPL = LastOpening->pl;
    if(fabs(LastOpening->DP) > DPL) /*Turbulence flow */
    {
      if(LastOpening->DP < 0.0)
        LastOpening->Flow = - LastOpening->FlowArea *
LastZone->SQRTDZ
                    *pow(-LastOpening->DP,LastOpening-
>FlowExponent);
      else
        LastOpening->Flow = LastOpening->FlowArea *
NearZone->SQRTDZ
                     *pow(LastOpening->DP,LastOpening-
>FlowExponent);
    }
    else  /* Laminar Flow */
    {
      LastOpening->Flow = LastOpening->DP * (LastOpening-
>ALaminar +
      fabs(LastOpening->DP) * LastOpening->BLaminar);
      if (LastOpening->DP < 0.0)
        LastOpening->Flow *= LastZone->SQRTDZ;
      else
        LastOpening->Flow *= NearZone->SQRTDZ;
    }
/* Calculte Derivative of Flow with respepect to DP */
    if(fabs(LastOpening->DP) > DPL)
    {
      DF = LastOpening->Flow * LastOpening-
>FlowExponent/LastOpening->DP;
    }
    else
    {
      DF = LastOpening->ALaminar
           + 2.0 * LastOpening->BLaminar *
fabs(LastOpening->DP);
      if(LastOpening->DP < 0.0)
        DF *= LastZone->SQRTDZ;
      else
        DF *= NearZone->SQRTDZ;
    }
/* Add Element Flow into first node to Zone Flows */
    BB[eqn_number[n-1]] += LastOpening->Flow;
    SUMAF[eqn_number[n-1]] += fabs(LastOpening->Flow);
/* Form Element Jacobina */
    nelem = 1;
    LM[0] = eqn_number[n-1];
    LM[1] = 0;
    KE[0][0] = -DF;
    KE[1][1] = 0.0;
    KE[0][1] = 0.0;
    KE[1][0] = 0.0;
    if(m > 0) /* m = 0 indicates exterior */
    {
/* Node 2 of element is building zone */
      nelem = 2;
      LM[1] = eqn_number[m-1];
      KE[1][1] = -DF;
      KE[0][1] = DF;
      KE[1][0] = DF;
/* Add element node 2 to Zone Flows */
      BB[eqn_number[m-1]] -= LastOpening->Flow;
      SUMAF[eqn_number[m-1]] += fabs(LastOpening->Flow);
    }
/* Add Element Jacobian to System Jacobian A */
    add_element_matrix(AA,PressureMatrixType,NZON,BandWidt
h,KE,LM,nelem);
  LastOpening = LastOpening->NextOpeningPtr;
  }
  if(LIST > 4) dump_matrix(AA,PressureMatrixType,NZON,Ban-
dWidth);
  if(LIST > 4) dump_vector(BB,NZON);
  factored = FALSE;
/* Solve for Pressure Correction */
  solve_linear_system(AA,BB,CC,&factored,Pressure-
MatrixType,
NumOfEqns,BandWidth);
/* Update Zone Pressures */  n = 0;
  if(LIST > 3) printf("\n    Initial ZonePressures \n");
  if(LIST > 3) fprintf(dataoutfile,"\n    Initial Zone-
Pressures \n");
  LastZone = FirstZone->NextZonePtr;
  while((LastZone != NULL) && (n < NZON))
  {
    LastZone->PZ -= CC[eqn_number[n]];
    if(LIST > 3 )
    {
      printf(" PZ: %4d %15.3e\n",n+1,LastZone->PZ);
      fprintf(dataoutfile," PZ: %4d
%15.3e\n",n+1,LastZone->PZ);
    }
    LastZone = LastZone->NextZonePtr;
    n++;
  }
/* Calculate Pressure Across Openings */
  LastOpening = FirstOpening;
  while(LastOpening != NULL)
  {
    n = LastOpening->AFSPTR[0];
    m = LastOpening->AFSPTR[1];
    LastOpening->DP = ZonePointer(m)->PZ - ZonePointer(n)-
>PZ
                      +LastOpening->PS + LastOpening->PW;
    LastOpening = LastOpening->NextOpeningPtr;
  }
/* Check Convergence */
```

Matrix Storage

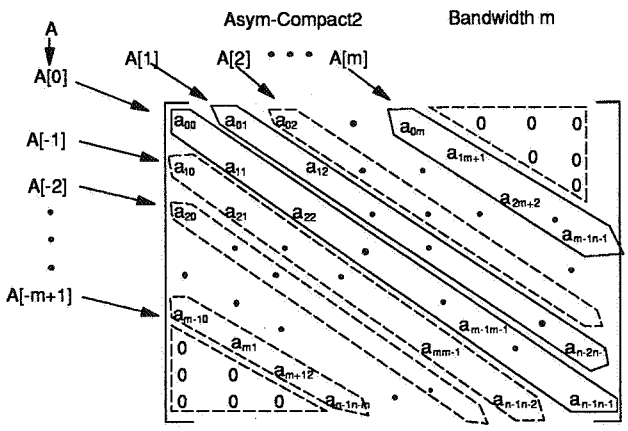
CONTAM88 uses seven types of matrix storage techniques: three of symmetric matrices and for asymmetric matrices. The asymmetric matrix storage types are : storage for the full N by N matrix (asymmetric_full), storage for a diagonally banded matrix by rows (asymmetric_compact) and storage for a diagonally banded matrix by arrays parallel to the diagonal (asymmetric_compact2) and a skyline storage technique. The symmetric matrix storage techniques are: storage for the N·(N-1)/2 diagonal and upper diagonal elements (symmetric_full), storage by arrays parallel to the diagonal by a diagonally banded symmetric matrix (symmetric_compact) and a skyline technique for symmetric matrices (skyline_symmetric). The above matrix storage techniques are used dynamically in **CONTAM88**.



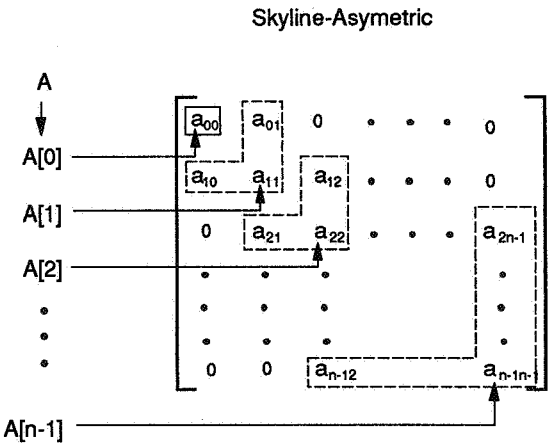
Asymmetric Full Matrix Storage Technique



Storage Technique Asymmetric Compact Matrix - Row

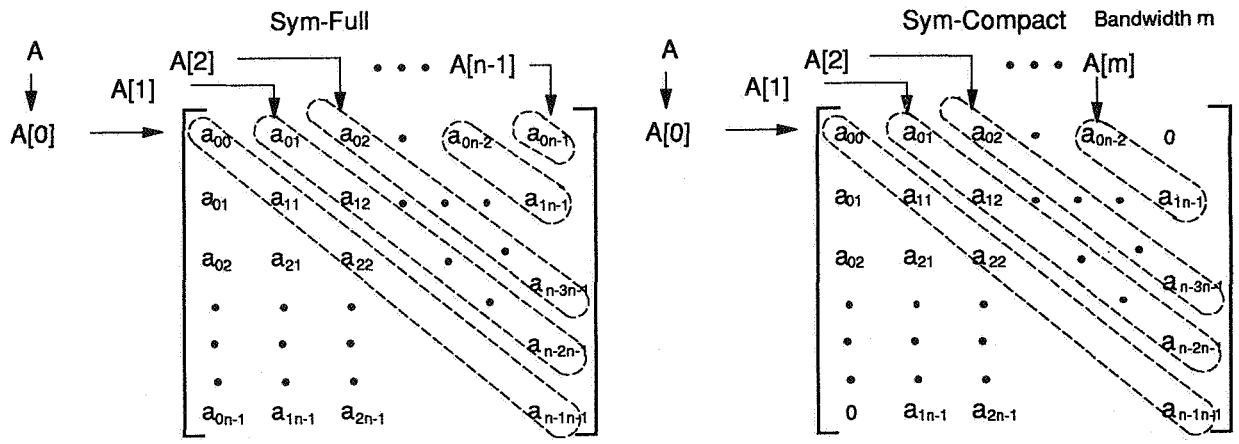


Asymmetric Compact Matrix - Diagonal Storage Technique

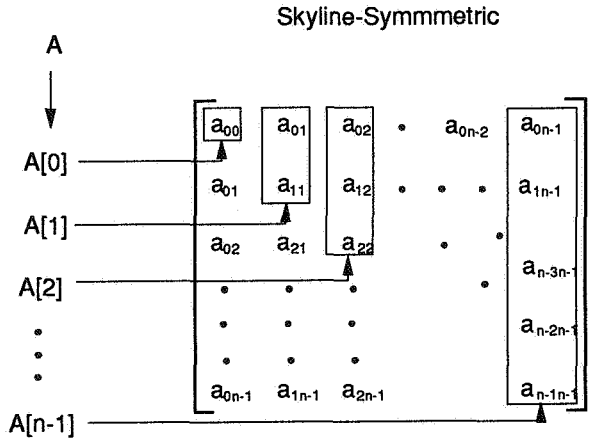


Asymmetric Skyline Matrix Storage Technique

Figure 13. Schematics of Asymmetric Matrix Storage Techniques



Symmetric Full Matrix Storage Techniques Symmetric Compact Matrix Storage Technique



Skyline Symmetric Matrix Storage Techniques

Figure 14. Schematics of Symmetric Matrix Storage Technique

The basic method for dynamically allocating for a matrix is a structured language such a "C" which has pointers as a data type is to allocate storage for an array of pointers which point to the storage for the one-dimensional arrays which make up the matrix and then allocate the storage for each of these one-dimensional arrays. To illustrate this technique consider defining the two data types vector and matrix for the storage of "double" variables.

```
typedef  double *vector
typedef  vector *matrix
```

where * in the pointer operator in "C". Then to allocate memory for a N⊗N matrix, first allocate storage for an array A of N "vectors" {A[0] ... A[N-1]} and then for each A[l] ,l=0...N-1 allocate storage for a linear array of N doubles. This can be accomplished with the "C" code (where error checking has been omitted);

```
int l;
matrix A;

A = (matrix) calloc(N,sizeof(vector));
for(l=0;l<N;l++) A[l] = (vector) calloc(N,si-
zeof(double));
```

Then either the notation *A[k][l]* or **(*(A+k)+l)* can be used to storage or retrieve the double variable stored in the k[th] row and l[th] column of the matrix. Figure 13 illustrates the meaning of these variables.

In many physical problems, the matrix describing the system contains many zeros and the non-zero elements exist near the diagonal. Such matrices are called banded (see figure 13). **CONTAM88** allows two types storage schemes for compact asymmetric matrices as illustrated in figure 14. The first stores data by rows, the second by one-dimensional arrays parallel to the diagonal. If m is the half bandwidth of the matrix

$$m = \max |i-j| \quad \ni A_{ij} \neq 0$$

then the code

```
int l,n1,n2,N,m; /* N = dimension, m = half bandwidth */
matrix A;

A = (matrix) calloc(N,sizeof(vector);
for(l=0;l<N;l++)
{
  n1 = (l-m > 0) ? l-m : 0;  /* max(l-m,0) */
  n2 = (l+m > N-1) ? N-1 : l+m ; /* min(l+m,N-1) */
  A[l] = (vector) calloc(n2-n1+1,sizeof(double));
  A[l] += n1; /* shift pointer to diagonal */
}
```

allocates the storage for the banded matrix and leaves the pointer A[l] pointing to the diagonal element. Leaving the pointer A[l] pointing to the diagonal element, in the author's opinion, produces cleaner code. Beware that in freeing the allocated memory, the pointer must be restored to its value when memory was allocated.

The value stored in A[i][i-j] is the non-zero value of the element in the i[th] row and j[th] column.

$$A_{ij} = A[i][i-j] \quad \text{if} \quad |i-j| \leq m$$

$$= 0 \quad \text{if} \quad |i-j| > m$$

The storage of a band asymmetric matrix by one-dimension arrays parallel to th diagonal can be accomplished by the code

```
int l,N,m; /* N = dimension, m = half bandwidth */
matrix A;

A = (matrix) calloc(2m+1,sizeof(vector);
A *= m; /* shift A to point to diagonal *
for(l=-m;l<m+1;l++)
```

```
{
  A[l] = (vector) calloc(N-abs(l),sizeof(double));
}
```

The value stored in A[j-i][min(i,j)] is the non-zero value of the element in the i^{th} row and j^{th} column.

$$A_{ij} = A[j-i][\min(i,j)] \quad \text{if} \quad |i-j| \le m$$

$$= 0 \quad \text{if} \quad |i-j| > m$$

Probably the most efficient storage scheme is the last one illustrated in figure 13. This is the skyline scheme. In this scheme one stores the non-zero elements in arrays A[l] which include the diagonal, the column of value to the last non-zero element above the diagonal and the row to right till the last non-zero element. If m[l] is the half bandwidth of this row/column

$$m[l] = \max_{\forall i < l}(l-i) \quad \ni \quad A_{kl} = 0 \quad \text{and} \quad A_{lk} = 0 \quad \forall k > i$$

The code for allocating memory for a skyline matrix is

```
int l,N,*m; /* N = dimension, m -> half bandwidth */
matrix A;

A = (matrix) calloc(N,sizeof(vector);
for(l=0;l<N;l++)
{
  A[l] = (vector) calloc(2*m[l]+1,sizeof(double));
  A[l] += m[l]; /* shift pointer to diagonal */
}
```

The value stored in A[max(i,j)][j-i] is the non-zero value of the element in the i^{th} row and j^{th} column.

$$A_{ij} = A[\max(i,j)][j-i] \quad \text{if} \quad |i-j| \le m[\max(i,j)]$$

$$= 0 \quad \text{if} \quad |i-j| > m[\max(i,j)]$$

Figure 14 illustrates the three storage schemes for symmetric matrices.

The scheme for the full symmetric matrix and the banded compact symmetric matrix both storage the upper triangular part of the matrix by one-dimensional arrays parallel to the diagonal. Both these methods can be allocate by the code:

```
int l,m,N; /* for a full symmetric matrix m = N-1 */
matrix A;

A = (matrix) calloc(m+1,sizeof(vector);
for(l=0;l<m+1;l++)
{
  A[l] = (vector) calloc(N-l,sizeof(double));
}
```

In these cases A[|i-j|][min(i,j)] is the value of the non-zero element in the i^{th} row and j^{th} column.

$A_{ij} = A[|\, j-i \,|][\min(i,j)]$ for a full symmetric matrix

and

$A_{ij} = A[|\, j-i \,|][\min(i,j)]$ if $|\, i-j \,| \le m$

$= 0$ if $|\, i-j \,| > m$ for a banded symmetric matrix

The code for allocating memory for a symmetric skyline matrix is

```
int l,N,*m; /* N = dimension, m -> half bandwidth */
matrix A;

A = (matrix) calloc(N,sizeof(vector);
for(l=0;l<N;l++)
{
  A[l] = (vector) calloc(m[l]+1,sizeof(double));
}
```

The value stored in A[max(i,j)][|j-i|] is the non-zero value of the element in the i^{th} row and j^{th} column.

$A_{ij} = A[\max(i,j)][|\, j-i \,|]$ if $|\, i-j \,| \le m[\max(i,j)]$

$= 0$ if $|\, i-j \,| > m[\max(i,j)]$

As it will be shown in the following sections, the bandwidths can be determined without forming the matrices by using the indices of the element nodes and the matrix solution and manipulation techniques of CONTAM88 preserves the structure of the matrices (that is, they do not destroy the bandwidth).

Solution of Linear System of Equations

The linear systems of equations formed by **CONTAM88** are solved using a LU decomposition following by forward and backward substitution of the factored matrix. This method preserves the matrix structures treated above: that is the solution technique requires only the values stored in the banded and skyline storage schemes. If one considers the linear set of equations

$$\overleftrightarrow{\mathbf{A}}\vec{\mathbf{X}} = \vec{\mathbf{F}}$$

then it is possible to factor the matrix $\overleftrightarrow{\mathbf{A}}$ into the product of a lower triangular matrix $\overleftrightarrow{\mathbf{L}}$ and an upper triangular matrix $\overleftrightarrow{\mathbf{U}}$

$$\overleftrightarrow{\mathbf{A}} = \overleftrightarrow{\mathbf{L}} \cdot \overleftrightarrow{\mathbf{U}}$$

where $\overleftrightarrow{\mathbf{L}} = \begin{bmatrix} 1 & 0 & 0 & 0 & \cdots & 0 \\ L_{21} & 1 & 0 & 0 & \cdots & 0 \\ L_{31} & L_{32} & 1 & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ L_{N-11} & L_{N-12} & \cdots & \cdots & 1 & 0 \\ L_{N1} & L_{N2} & \cdots & \cdots & L_{NN-1} & 1 \end{bmatrix}$

and $\overleftrightarrow{\mathbf{U}} = \begin{bmatrix} U_{11} & U_{12} & U_{13} & U_{14} & \cdots & U_{1N} \\ 0 & U_{22} & U_{23} & U_{34} & \cdots & U_{2N} \\ 0 & 0 & U_{33} & U_{34} & \cdots & U_{3N} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & 0 & \cdots & U_{NN} \end{bmatrix}$

The decomposition can be accomplished by the algorithm

$$U_{11} = A_{11}$$

for $s = 2, \ldots, N$

$$L_{si} = \left(A_{si} - \sum_{m=1}^{i-1} L_{sm} U_{mi} \right) \frac{1}{U_{ii}} \quad i = 1, \ldots, s-1 \tag{47}$$

$$U_{is} = A_{is} - \sum_{m=1}^{i-1} L_{im} U_{ms} \quad i = 1, \ldots, s \tag{48}$$

Note that in performing the LU decomposition of a matrix A, the results can be stored in the same storage locations as the original matrix $\overleftrightarrow{\mathbf{A}}$ if the 1's on the diagonal of lower triangular matrix $\overleftrightarrow{\mathbf{L}}$ are not stored.

$$\overleftrightarrow{\mathbf{A}} = \begin{bmatrix} U_{11} & U_{12} & U_{13} & U_{14} & \cdots & U_{1N} \\ L_{12} & U_{22} & U_{23} & U_{24} & \cdots & U_{2N} \\ L_{31} & L_{32} & U_{33} & U_{34} & \cdots & U_{2N} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ L_{N1} & L_{N2} & L_{N3} & \cdots & L_{NN-1} & U_{NN} \end{bmatrix}$$

using the algorithm

for $s = 2, \ldots, N$

$$A_{si} = \left(A_{si} - \sum_{m=1}^{i-1} A_{sm} A_{mi} \right) \frac{1}{U_{ii}} \quad i = 1, \ldots, s-1 \tag{49}$$

$$A_{is} = A_{is} - \sum_{m=1}^{i-1} A_{im} A_{ms} \quad i = 1, \ldots, s \tag{50}$$

The LU decomposition preserves the banded and symmetric structure of the matrices formed in **CONTAM88**[10].

5 Adsorption Modeling

Adsorption-desorption mass transport processes introduce complexities in contaminant dispersal analysis that demand thoughtful structuring of both the data describing the building system to be analyzed and the associated systems of equations that describe the dispersal of contaminants in the building. To set the stage for consideration of these complexities, the underlying theory of one of two families of adsorption elements, that have been recently developed to augment the existing library of contaminant dispersal elements[11], will be reviewed here.

Adsorption involves the separation of a substance _ the adsorbate _ from one phase, indoor air in the present context, and the accumulation of that substance on the surface of another phase _ the adsorbent _ building materials and furnishings, here. Limiting consideration to those adsorbate-adsorbent systems that exhibit reversible sorption (i.e., both adsorption and desorption transport processes are possible) the adsorption process may be thought to be a reversible exchange between alternative phases of the adsorbate _ the free phase and the adsorbed phase. In the building context such a reversible adsorption process may be idealized as illustrated below in Figure 15.

Air flows into a given building zone carrying, perhaps, an adsorbate species _ identified here with the Greek leading superscript α. Within the core of the building zone the adsorbate species exists at a bulk mean concentration of ${}^{\alpha}C$ (mass-a/mass-air) and is transported via diffusion transport processes to a near-surface location where its free phase concentration is ${}^{\alpha}C^{*}$. At this near-surface location adsorption processes come into play to bind the species to the surface establishing the adsorbed surface-phase concentration ${}^{\alpha}C_{s}$ (mass-a/mass-adsorbent). The rate of species transport to and from the adsorbent surface is, thus, determined by both diffusion and adsorption mechanisms.

Diffusion: Diffusion transport to and within effective adsorbents invariably involves a complex variety of transport mechanisms including boundary layer molecular and turbulent diffusion from the bulk phase and molecular, Knudsen, Poiseuille, and surface diffusion within macroporous and microporous interstices[12]. For our present purposes it will be sufficient to limited consideration to boundary layer diffusion which may be modeled as a spatially discrete process using boundary layer

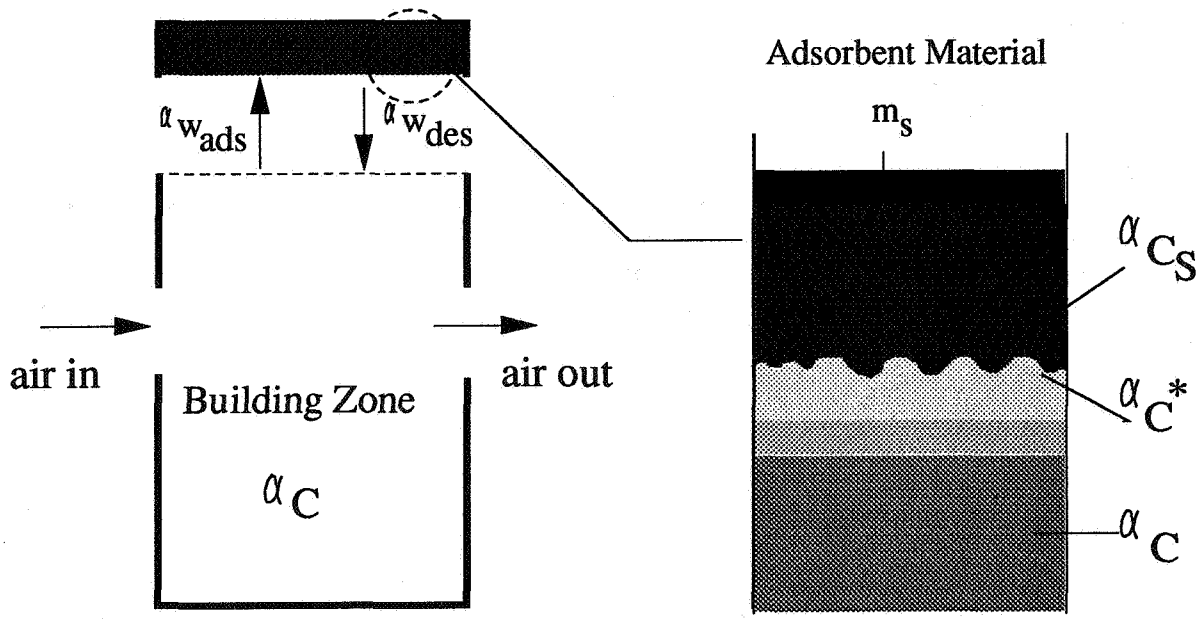


Figure 15. Adsorption Element Variables

theory[13]; macro and microporous diffusion processes which are, by their nature, spatially continuous may, conceivably, be modeled using discrete finite element approximations. From boundary layer theory, the net mass transport of the adsorbate species, $^{\alpha}w_S$ (mass-a/time), from the bulk phase to the near-surface location is directly related to free-phase concentration difference between the bulk and near-surface locations as:

$$\alpha_{ws} = (^{\alpha}w_{ads} - ^{\alpha}w_{des}) = h_m \rho A_S (^{\alpha}C - ^{\alpha}C^*) \quad (51)$$

where h_m (length/time) is the average film mass transfer coefficient acting over the adsorbent surface, r is the film density of air, the average of the bulk and near-surface densities, and A_S is the projected surface area of the adsorbent. The film transport coefficient is correlated to fundamental characteristics of the airflow over the adsorbent; the surface flow Reynolds number and the free phase Schmidt number.

Adsorption: In typical circumstances the rate of the adsorption step may be expected to be practically instantaneous relative to the diffusion transport processes and/or the rate of mass transport by advection into a given building zone, thus, it is not normally necessary to explicitly model the adsorption kinetics. Although practically instantaneous, adsorption is limited by its equilibrium state. That is to say, for closed systems under steady conditions the rate at which adsorbate molecules bind to the adsorbent surface will eventually equal the rate at which they are released from the surface and the concentration in the free and adsorbed phases will remain constant at their respective

equilibrium values, ${}^{\alpha}C^{*}_{e}$ and ${}^{\alpha}C_{Se}$. In functional notation, we may say that the adsorbed phase equilibrium concentration is related to the free phase equilibrium concentration and the thermodynamic state of the system defined by temperature, T, and pressure, P, as:

$$ {}^{\alpha}C_{Se} = f({}^{\alpha}C^{*}_{e}, T, P) \tag{52} $$

These equilibrium relations are, generally, unique to the adsorbate-adsorbent system being considered and, when reported for isothermal conditions at atmospheric pressure, are identified as adsorption isotherms. Although a large variety of models have been proposed for adsorption isotherms three physically-based models, the Linear, Langmuir and BET models, have become fundamental to the field. The Linear model may be expected to useful in those situations where adsorbate free-phase concentrations are low (i.e., relative to saturation levels) and it is especially straightforward:

$$ {}^{\alpha}C_{Se} = K_P {}^{\alpha}C^{*}_{e} \tag{53} $$

where K_P is the so-called partition coefficient that exhibits an Arrhenius type dependency on temperature for ideal reversible systems.

Adsorption Element Equations: Two families of adsorption element equations may be formulated by: a) demanding instantaneous conservation of species mass in the adsorbent phase; b) demanding equilibrium conditions prevail at the surface of the adsorbent (i.e., as described by one of the several equillibrium relations available); and c) modeling the diffusion transport step as either instantaneous _ equillibrium adsorption _ or as described by the boundary layer theory presented above _ boundary layer diffusion controlled adsorption. These element equations have the general form:

$$ [{}^{\alpha}\mathbf{w}^{e}]\{{}^{\alpha}\mathbf{C}^{e}\} + [{}^{\alpha}\mathbf{m}^{e}]\frac{d\{{}^{\alpha}\mathbf{C}^{e}\}}{dt} = \{{}^{\alpha}\mathbf{g}^{e}\} \tag{54} $$

where $\{{}^{\alpha}\mathbf{C}^{e}\}$ is a vector of contaminant species a discrete concentration variables associated with the element "e" and $\{{}^{\alpha}\mathbf{g}^{e}\}$ is a vector of element derived contaminant species generation rates. The element matrices $[{}^{\alpha}\mathbf{w}^{e}]$ and $[{}^{\alpha}\mathbf{m}^{e}]$ are square transformation matrices known as the element species transport matrix and species mass matrices respectively.

For the combination of the linear equillibrium adsorption model and the boundary layer diffusion model, a two-node element results with:

$$ \{{}^{\alpha}\mathbf{C}^{e}\} = \begin{Bmatrix} {}^{\alpha}C_{i} \\ {}^{\alpha}C_{j} \end{Bmatrix} \quad ;\text{i = well-mixed zone node, j = adsorbent node} \tag{54} $$

$$ [{}^{\alpha}\mathbf{m}^{e}] = \begin{bmatrix} 0 & 0 \\ 0 & m_S \end{bmatrix} \tag{54} $$

$$ [{}^{\alpha}\mathbf{w}^{e}] = (h_m \rho A_S) \begin{bmatrix} 1 & -\dfrac{1}{K_P} \\ -1 & \dfrac{1}{K_P} \end{bmatrix} \tag{55} $$

where m_s is the total adsorbent mass available.

In conclusion then, when diffusion transport is accounted for, adsorption modeling will demand the introduction of additional degrees of freedom (i.e., nodes), beyond those corresponding to each well-mixed zone, to be included in the system model. In multi-contaminant dispersal analysis contaminants may be selectively adsorbed, thus, in general, the number of additional adsorbent nodes associated with each building zone may be expected to vary in number and type from zone to zone A data structure to include material absorption in a general contaminant dispersal analysi is shwon in the following figure. One adds to the data structure of the zone data of CONTAM88 a pointer to a list of data structures which contain the absorption characteristics of the materials contained in each zone. The contaminant dispersal assembly process then adds the corresponding material nodes and coupling to the zone contaminant dispersal equations. This theory and assembly process has been implemented in CONTAM89, a prototype of the next version of CONTAM88.

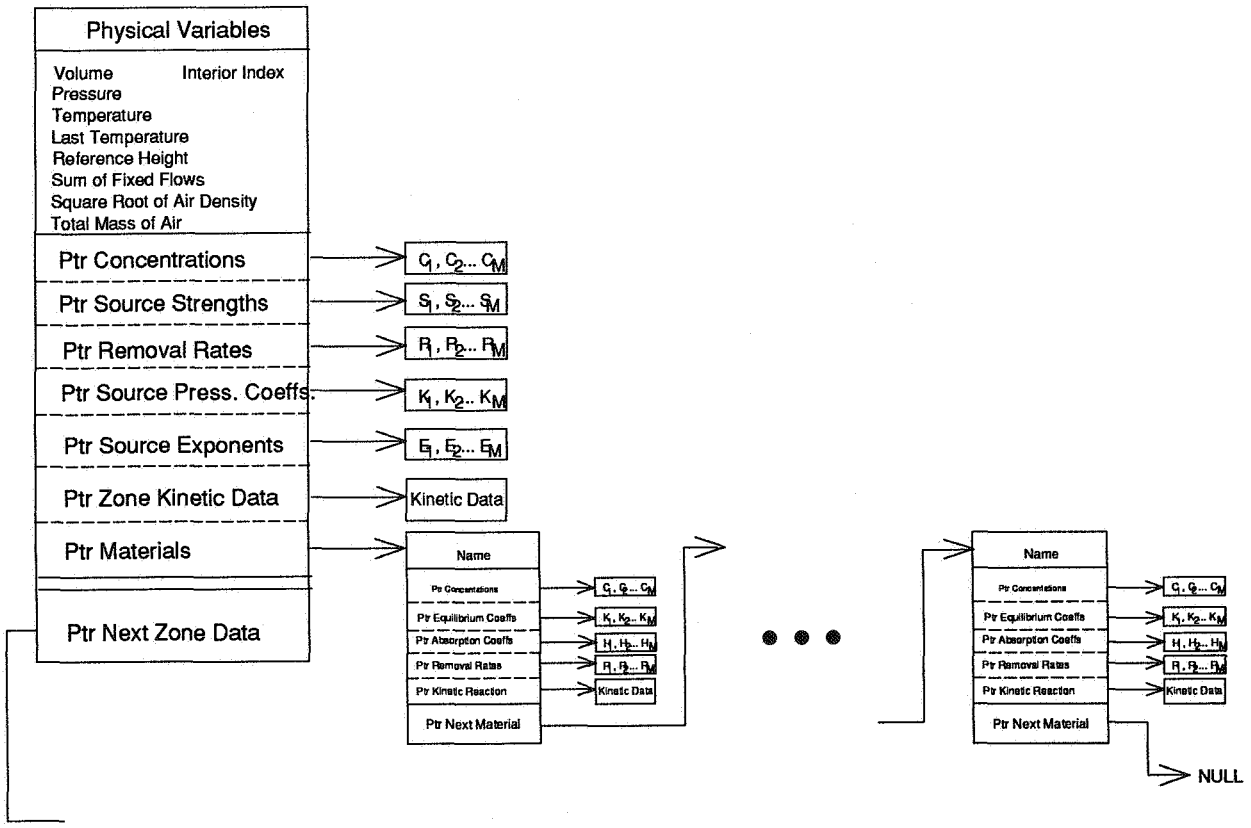


Figure 16. Schematic of Zone data Structure to Include Material Absorption

6 Modeling of a 12 Storey Apartment Building

NBSAVIS and **CONTAM88** has been used to model the air flows and contaminant dispersal in a 12 storey apartment building shown schematically in figures 17 and 18.

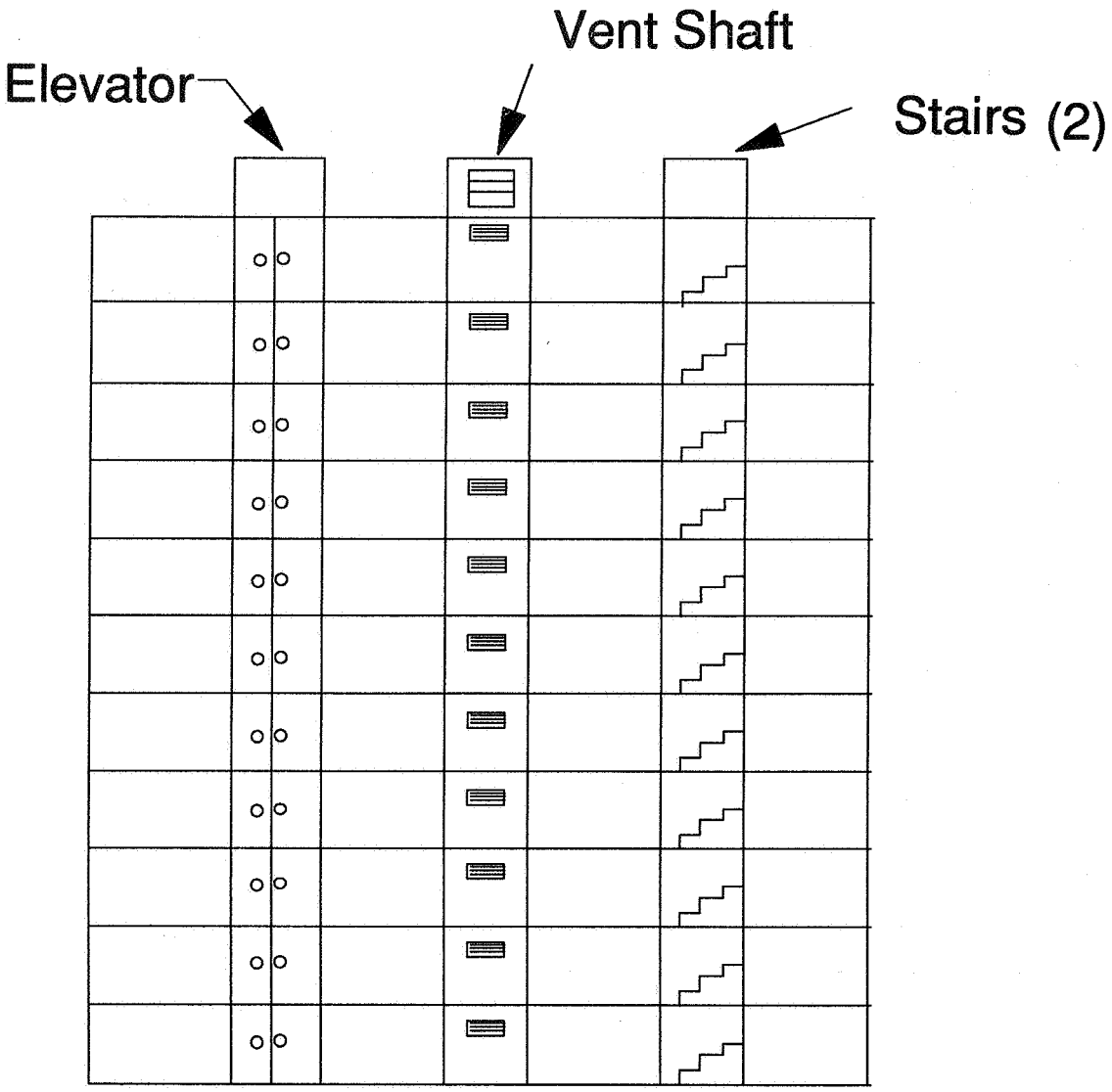


Figure 17. Schematic of the Elevation of a 11 Storey Apartment Building

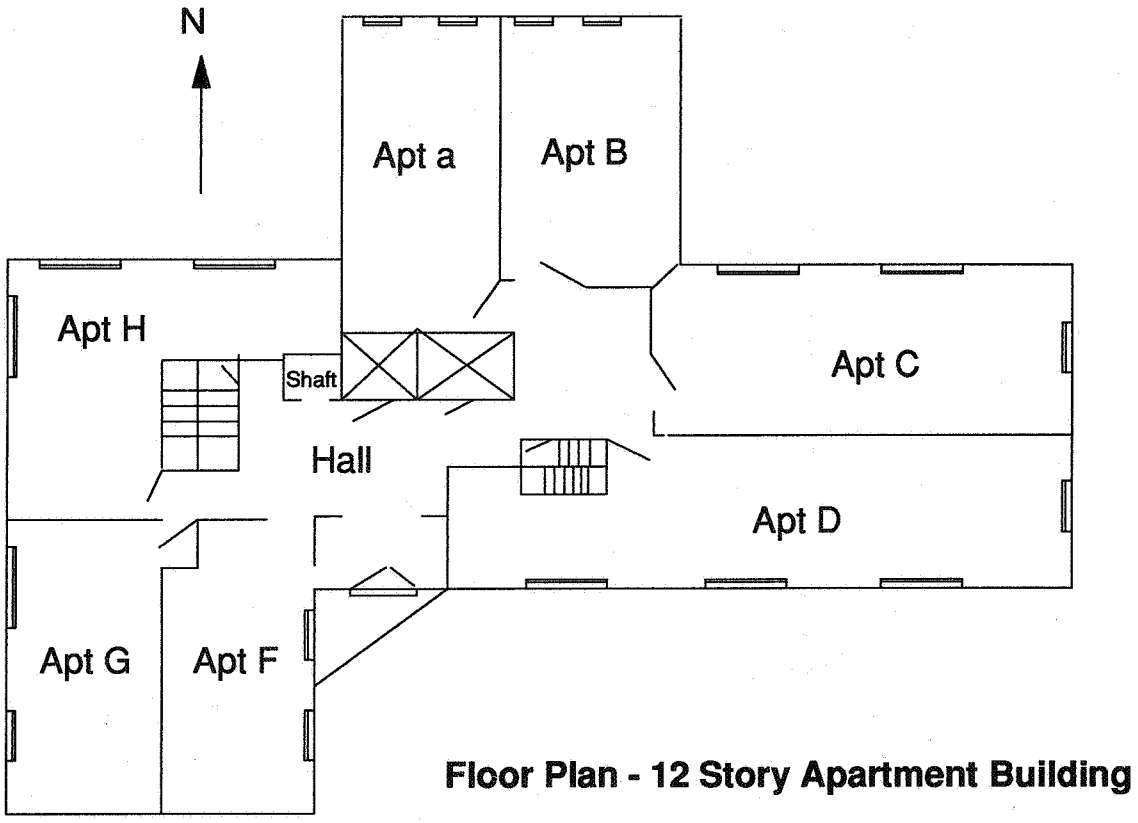


Figure 18. Typical Floor Plan of the Apartment Building

The apartment building has two stairwells running from the main floor to the roof. It also has an elevator shaft and a passive ventilation shaft. There is a large grill in the entrance way of the first floor to provide air for the ventilation shaft. Each floor has a hallway and 7 apartments. The building was described by NBSAVIS as a 100 room building: the two stairwells, elevator shaft and the ventilation shaft plus for each floor, the hall and each apartment. The description produced by **NBSAVIS** had over 1000 flow paths. The results of simulating a pressure driven radon source in the apartments on the first floor are shown in figure 19. It took approximate 20 seconds and 5 iterations to solve the flow-pressure system of equations on a 386/33 IBM clone without a math coprocessor and 2 seconds to solve the steady state contaminant dispersal system of equations. The problem required about 40K of memory by **NBSAVIS** for the building description and 180K (leaving 190K of the 640K of memory not used) by **CONTAM88** for solving the flow and contaminant dispersal systems.

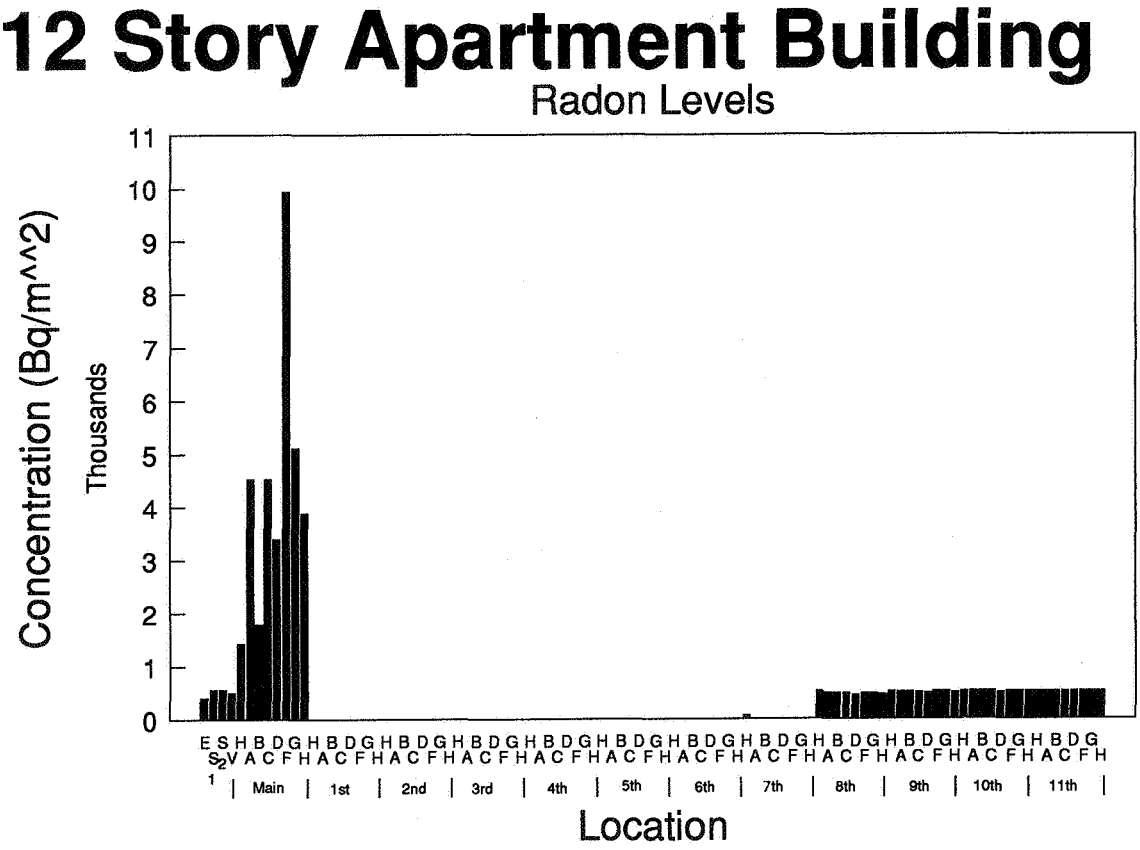


Figure 19. Results of Simulating Radon Migration in a 12 Storey Building

Figure 19 shows the high radon levels on the first floor in the apartments where the radon sources are. However it also shows that radon will migrate up the shafts of the building and exit through the apartments on the upper levels due to the stack effect. Note that there is no radon on floors 1 through 7 since there is no source of radon on the upper floors and the air flow enters the apartments on these floors form the exterior and exists through the shafts. On floors 8 through 11, the air flow is from the shafts through the apartments to the exterior. Level of radon in the apartments on floor 8 through 11 is basically the level of radon in the vertical shafts of the building.

References

1. Walton, G.N. "A Computer Algorithm for Estimating Infiltration and Inter-Room Air Flows", February 1983, NBSIR 83-2635

2. Walton, G.N. "A computer algorithm for predicting infiltration and interroom airflows", 1984, ASHRAE Transactions, vol 90,Part 1

3. Persily, A.K. & Grot, R.A., "The Airtightness of Office Building Envelopes", ASHRAE SP49, Thermal Performance of Exterior Envelopes of Buildings III, 1986

4. Axley, J., "Indoor Air Quality Modeling Phase II Report", 1987, NBSIR 87-3661

5. Axley, J., "Progress Toward A General Analytical Method for Predicting Indoor Air Pollution in Buildings - Indoor Air Quality Modeling Phase II Report", 1988, NBSIR 88-3814

6. Liddament, M.W, "Air Infiltration Calculation Techniques - An Applications Guide", Air Infiltration and Ventilation Centre, Coventry, UK, 1986

7. Nazaroff, W. W & Nero, A.V. (Editors), Radon and Its Decay Products in Indoor Air, John Wiley & Sons, NY,NY, 1988.

8. ASHRAE Handbook of Fundamentals, ASHRAE, Atlanta, GA, 1988.

9. Walton, G.N., "AIRNET - A Computer Program for Building Airflow Network Modelling", NISTIR 89-4072. 1989

10 Grot, R.A., User's Manual for NBSAVIS and CONTAM88, NISTIR 90-XXXX (in review)

11 Axley, J. W. Adsorption Modeling for Macroscopic Contaminant Dispersal Analysis. NIST/GCR 90/573. National Institute of Standards and Technology. 1990.

12. Ruthven, D. M. "Adsorption Kinetics." Adsorption Science and Technology. Rodrigues ed. 1989 Kulwer Academic Publishers. Dordrecht.

13. White, F. M. "Heat and Mass Transfer." 1988 Addison-Wesley Pub. Co. New York.

Discussion

Paper 13

Earle Perera (BRE, UK)

Has your air movement model been validated against a known bench-mark case? If not, then how can all those who have similar multizone programs get together to set-up such a validation?

R.Grot (Lagus Applied Technology, USA)

These models have not been completely validated. The numerical procedures have been checked quite extensively by George Wilton. Andy Persily and I wrote a paper for ASHRAE comparing the results of AIRMOV on earlier version with two large building Air Infiltration results; however the accuracy of the multizone flows has not been checked. The last part of your question is more political and depends on the ability of many of us to convince our countries to fund validation research.

C-A Roulet (LESO, Switzerland)

Evaluation of computer codes is a huge task which consists not only in comparing computation results with measurements but also in evaluating the effects of small changes in input data on the results and this is all the possible range of the input parameters. I hope that Annex 23 can make some progress in this quite unexplored domain.

R.Grot (Lagus Applied Technology, USA)

I also hope that Annex 23 can provide answers to this very important issue.