IMPROVING SPEED AND ROBUSTNESS OF THE COMIS SOLVER

D.M. Lorenzetti and M.D. Sohn

Indoor Environment Department
Lawrence Berkeley National Laboratory
Berkeley CA 94720, USA
Report LBNL-44792

Published in
Proceedings of the 7th International Conference on Air Distribution in Rooms (RoomVent 2000) v.1 (2000) 241–246

ABSTRACT

The numerical investigation of airflow and chemical transport characteristics for a general class of buildings involves identifying values for model parameters, such as effective leakage areas and temperatures, for which a fair amount of uncertainty exists. A Monte Carlo simulation, with parameter values drawn from likely distributions using Latin Hypercube sampling, helps to account for these uncertainties by generating a corresponding distribution of simulated results. However, conducting large numbers of model runs can challenge a simulation program, not only by increasing the need for fast algorithms, but also by proposing specific combinations of parameter values that may define difficult numerical problems.

The paper describes several numerical approaches to improving the speed and reliability of the COMIS multizone airflow simulation program. Selecting a broad class of algorithms based on the mathematical properties of the airflow systems (symmetry and positive-definiteness), it evaluates new solution methods for possible inclusion in the COMIS code. In addition, it discusses further changes that will likely appear in future releases of the program.

KEYWORDS

Airflow network, Line search, Newton–Raphson, Multizone, Simulation, Trust region.

INTRODUCTION

The numerical investigation of a building's airflow and chemical transport characteristics involves identifying values for model parameters, such as effective leakage areas and temperatures, for which a fair amount of uncertainty and variability exist. When seeking to represent a general class of building, rather than a particular building for which design information and measured data may exist, the uncertainties become greater still. For example, as part of an effort to develop guidelines for building managers seeking to respond to indoor pollutant releases, Sohn et al. (1998) modeled a five-story office building using the multizone airflow simulation program COMIS (Feustel 1999). The model, meant to typify intermediate-size, open-style commercial spaces, identified 13 critical parameters describing the structure and its use. With the parameter values varying widely among sample spaces, clearly a single simulation cannot represent the whole range of possible behaviors.

Monte Carlo simulation provides one approach to characterizing uncertainties in the model predictions. Assigning a likely distribution to each critical parameter defines a corresponding distribution of possible building representations. Discretizing this distribution using Latin Hypercube sampling (Iman et al. 1980) yields a set of simulations, and a corresponding range of possible flow characteristics, for the building under study. Thus Sohn et al., sampling from assumed distributions of their 13 model parameters, generated 2000 specific COMIS simulations. The response of each to a pollutant release indicated, in aggregate, the uncertainty expected in a real building, about which little may be known other than that it belongs to the general class of buildings defined earlier.

Unfortunately, this technique burdens the simulation tool: first by requiring a large number of runs, and hence increasing the total execution time; second by increasing the chance that some particular combination of parameter values will define a difficult numerical problem. For example, in initial tests COMIS completed 1825 out of 2000, or about 91%, of the simulations required to characterize the five-story office.

This paper describes changes to COMIS v3.0, that begin to address these problems. Because of known nonconvergence associated with duct junctions (Feustel 1999), the investigation initially focused on methods for stabilizing the solution algorithm. However, efforts to recode the solver showed that the initialization scheme employed at each time step was largely responsible for slow and nonconverging simulations. The sections below discuss these issues, and show how relatively modest changes to the solver can improve its execution speed. At least some of these changes should appear in COMIS v3.1 (the Berkeley Laboratory web site http://epb1.lbl.gov/comis/ provides links to the COMIS code).

BUILDING AIRFLOW SYSTEMS

Multizone airflow models, such as COMIS, represent zones (e.g., rooms and duct junctions) as nodes of unknown pressure, connected via discrete flow paths (such as doors, cracks, and duct-work) with unknown mass flow. The governing equations describe steady-state mass conservation at the nodes, and the pressure-flow relations of the paths. COMIS adopts a nodal formulation of the problem, treating the node pressures as independent, and updating the flows to keep them in agreement with the current pressure estimates. Walton (1989) details nodal models. Wray & Yuill (1993) discuss this and other possible formulations.

The nodal formulation requires pressure-flow relations that express the flow, $f_{i\text{-}j}$, through the element(s) connecting node i to node j as a function of their pressures, P_i and P_j. In a useful idealization, the flow depends only on the pressure drop between the nodes:

$$f_{i\text{-}j} = f_{i\text{-}j}\{P_i - P_j\} \ . \tag{1}$$

The flow element pressure drop includes wind and thermal buoyancy effects (Feustel 1999). If every pressure-flow relation follows Eqn. 1, then at least one node (typically that representing the building's surroundings) must have its pressure fixed.

The solution of the nodal equations proceeds iteratively. For each guess at the unknown pressures, the program sums the flows entering each variable-pressure node, then adjusts the pressures, seeking to enforce mass balance. Call the pressure vector at the k^{th} iteration $P_{[k]}$, and the corresponding sums of flows $r_{[k]}$. Then $r_{[k]}$ gives a vector of residuals that the solver seeks to zero. Most nonlinear solvers use some variation on Newton–Raphson's method to zero the residuals (Dennis & Schnabel 1996). This method finds the Jacobian matrix, $J_{[k]}$, of derivatives of the residual vector, forms a local model of the residuals using those derivatives, and calculates the next set of pressures in order to zero that model:

$$P_{[k+1]} = P_{[k]} - J_{[k]}^{-1} r_{[k]} \ . \tag{2}$$

When the flow elements obey Eqn. 1, they yield a symmetric Jacobian. If in addition every flow element has $\partial f_{i\text{-}j}/\partial P_i \geq 0$, then the Jacobian is positive-definite (Axley 1989); note that Axley proves a stronger result than found in most of the building airflow literature. Since a symmetric positive-definite matrix cannot be singular, and may be factored without pivoting (Dennis & Schnabel 1996), flow elements of this form simplify the numerics of an equation solver considerably.

INITIALIZATION MODIFICATIONS

Each time step of a simulation requires an initial $P_{[1]}$ from which to start the iterative solution. COMIS v3.0 finds $P_{[1]}$ based on two user-specified inputs (Feustel 1997). Nominally the control flag UseOPz determines how the program establishes initial pressures, while NoInit determines whether or not to apply linear initialization before attempting the fully nonlinear problem. Linear initialization replaces each flow element relation with a straight-line approximation meant to represent its global behavior; see Walton (1989). With both flags set to their default values of zero,

COMIS begins every time step by adjusting the zone pressures to counter thermal buoyancy, seeking a zero net pressure drop across each flow element. After setting these pressures, by default the program performs linear initialization.

As noted above, this default initialization causes about 9% of the parameterized office building simulations to fail. In all failed cases, the program completes the first time step successfully, but fails at later simulated times. A better initialization scheme, achieved by setting UseOPz = 1, begins each new time step with $P_{[1]}$ set to the final pressures calculated at the preceding step. This allows COMIS to complete all 2000 simulations successfully. However, inspecting the source code reveals that setting UseOPz = 1 prevents linear initialization at the first time step, and in fact prevents NoInit from having any effect at any time step.

A second problem with UseOPz concerns its overloaded use for controlling density updates. By default, the solver fixes zone densities at values consistent with the initial pressures, $P_{[1]}$. Setting UseOPz = 2, or specifying LoopRho in the &-Pr-Simu section of the input file, nominally causes the solver to update zone densities with every pressure iteration. This forces UseOPz to control two unrelated aspects of the program: pressure initialization and density updates. Thus setting UseOPz = 0 or 1 effectively cancels an earlier request for density updates via the LoopRho option.

Inspecting the source code also shows that the linear initialization stage, if invoked, runs iteratively. That is, instead of taking a single Newton–Raphson step using the global linearization, it seeks to solve that linearized system exactly, using a series of Newton–Raphson steps. Unfortunately, not every flow element has a global linearization defined. This means the program can spend many iterations solving, or attempting to solve, a partly nonlinear system that bears only marginal resemblance to the nonlinear system of interest.

To address these problems, we updated COMIS to initialize the first time step according to a control flag Stp1Init, and to initialize subsequent time steps according to Stp2Init. Both flags may either zero the pressure drops (value 0), or zero the drops and then take a single step on the globally linearized system (value 1). In addition, setting Stp2Init = 2 initializes subsequent time steps using the pressures calculated at the previous time step. These two flags replace UseOPz and NoInit, respectively, in the new &-Pr-Cont section of our input files. A new internal variable handles density updates as specified by the LoopRho option.

SOLVER MODIFICATIONS

At each time step, after any pressure initialization, the program turns to the fully nonlinear system. Close to a solution, the pure Newton–Raphson iteration defined by Eqn. 2 converges quickly, but it may diverge if the step to $P_{[k+1]}$ exceeds the range over which the derivative information in the Jacobian remains valid (Dennis & Schnabel 1996). For airflow systems containing highly nonlinear elements, such as crack elements, duct fittings, and large openings, Newton–Raphson may fail, no matter how many iterations it is allowed to take (Herrlin & Allard 1992, Feustel 1999).

For stability, COMIS uses a damping factor, μ, to damp the Newton–Raphson step:

$$P_{[k+1]} = P_{[k]} - \mu J_{[k]}^{-1} r_{[k]} \ . \tag{3}$$

Setting $\mu = 1$ recovers the original method, while $0 < \mu < 1$ gives the same search direction, but places $P_{[k+1]}$ at a shorter step. Recall that Newton–Raphson zeros the residual models formed using the local derivatives stored in $J_{[k]}$. Since this derivative information remains valid for short enough steps about $P_{[k]}$, and since the symmetric positive-definite Jacobian always admits inversion, then in exact arithmetic it must be possible to solve the system by repeated iterations with small enough μ.

In general, an algorithm should pick a relaxation factor small enough to avoid instability in the pressure iterates, but large enough to minimize the number of iterations required. Wray & Yuill (1993) found that a constant relaxation factor of 0.75 works well for many airflow systems; the current versions of both the COMIS and CONTAM airflow simulation programs use a variation on this idea, described by Walton (1997), to choose from among a fixed set of relaxation constants, depending on the change in the mass balances from iteration to iteration.

The mathematical literature defines several mature methods for selecting the damping factor in Eqn. 3 (Dennis & Schnabel 1996). These line search algorithms apply optimization theory to the

TABLE 1
ITERATIONS REQUIRED, BY SOLVER AND INITIALIZATION SCHEME

Input file	Std solver (5)		New solver (6)		Input file	Std solver (5)		New solver (6)	
	Zero (0)	Lin (1)	Zero (0)	Lin (1)		Zero (0)	Lin (1)	Zero (0)	Lin (1)
a01	10	18	16	15	b10	4	14	5	4
a02	7	19	18	13	b11	4	14	5	4
a03	8	11	3	6	b12	3	11	3	3
b01	2	2	2	2	b13	4	13	4	4
b02	3	5	3	3	b14	5	13	5	4
b03	2	8	2	3	b15	8	15	6	5
b04	4	8	4	3	b16	8	15	6	5
b05	4	13	5	4	b17	4	9	5	3
b06	4	6	12	4	b18	4	9	5	3
b07	10	12	6	4	b19	4	9	5	3
b08	4	10	4	4	b20	3	8	4	4
b09	5	10	5	3					

problem of selecting μ, by forming a scalar cost function from the residual vector, and then seeking a minimum of that cost function. The sum of squares of the residuals, r-square, provides a suitable cost function. Not only does it have a global minimum at $r = 0$, the solution of the nonlinear system, but any other minima can occur only at singularities in the Jacobian. Since the positive-definite Jacobian has full rank, in exact arithmetic an airflow system solver can always take a step that reduces the cost function.

We programmed a trust region based line search algorithm, adapted from Dennis & Schnabel (1996), as a new COMIS airflow solver (SLVSEL = 6). Like all minimization methods, the algorithm reduces the cost function at every iteration. In case it tries an overly long step, and fails to decrease r-square, it tries again with smaller μ. Thus the algorithm can take multiple residual evaluations at a single iteration. A trust region algorithm updates the expected length of a successful step from one iteration to the next, expanding and contracting the trust length depending on how well the actual value of r-square matches the predicted result. Our method follows the published one fairly closely, except that in order to avoid the possibility of stagnation due to numeric effects, it imposes the constraint $\mu \geq 10^{-6}$.

RESULTS

Testing the trust region algorithm on a number of COMIS simulations shows that it is generally competitive with the standard solver (SLVSEL = 5), provided each solver uses the appropriate initialization scheme. For 23 COMIS input files, Table 1 records the number of iterations each solver requires to complete the first time step, initializing with either zero pressure drops (STP1INIT = 0) or linear initialization (STP1INIT = 1). The iteration count includes the one associated with linear initialization, if used. The input files labeled "a-" come from the office simulations of Sohn et al., while those labeled "b-" are standard COMIS test cases.

The results show that both solvers are sensitive to the initialization method, with the standard solver taking many more iterations on average, and the new solver taking marginally fewer iterations, when using linear initialization. The significantly greater iteration counts for the standard solver under linear initialization may indicate that one or more flow elements have unreasonable global linearizations, but this remains for later investigation. Inspecting the solution trajectories for the trust region method shows that zeroing the pressure drops, which starts the flow elements in a regime of operation where small changes in the pressures dramatically change the slope of the pressure-flow curve, causes the algorithm to take a very short step in its first iteration. The solver subsequently increases the trust length, but not as aggressively as it might. The trust region method sacrifices speed for stability, if necessary, and when the stability problems arise from e.g. a crack model

operating near zero pressure drop, this may be to its detriment. Linear initialization, by jumping the solution away from these regimes of operation, avoids this behavior.

Comparing the new solver, under linear initialization, to the standard method, initialized with zero pressure drops, shows that the trust region method usually performs as well as, or slightly better than, the standard COMIS solver. However, none of the simulations shown in the table represent especially difficult problems, so one would not expect to see great differences between the two solvers. For more difficult problems, for example ones with duct fittings, we expect the trust region based solver to perform more robustly than the standard COMIS routine, due to its ability to adaptively choose finer gradations in the relaxation parameter. Again, this improved reliability will exact a toll in iteration counts on some problems.

CONCLUSIONS

A few relatively simple programming changes can improve the performance of the COMIS solver, and at the same time enhance the user's control of the initialization scheme. A trust region based solver performs well in the limited tests described, however, we have not challenged either solution algorithm with particularly difficult simulations. The authors invite the submission of COMIS input files known to cause the standard solver to fail, in order to further test and refine the algorithms.

The numerical experiments described here do not pursue the question of convergence difficulties associated with duct junctions, which remains open for further investigation.

ACKNOWLEDGEMENTS

This work was supported by the Office of Nonproliferation Research and Engineering, Chemical and Biological National Security Program, of the National Nuclear Security Administration under U.S. Department of Energy Contract No. DE-AC03-76SF00098.

REFERENCES

Axley J.W. (1989). Multi-Zone Dispersal Analysis by Element Assembly, *Building and Environment* **24:2**, 113-130.

Dennis J.E., Jr. and Schnabel R.B. (1996). *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Society for Industrial and Applied Mathematics, Philadelphia PA, USA.

Feustel H.E. and Smith B.V. (1997). *COMIS 3.0 User's Guide*, available from the Lawrence Berkeley Laboratory, Berkeley CA, USA. Download from http://epb1.lbl.gov/comis/.

Feustel H.E. (1999). COMIS—an International Multizone Air-Flow and Contaminant Transport Model. *Energy and Buildings* **30:1**, 3-18.

Herrlin M.K. and Allard F. (1992). Solution Methods for the Air Balance in Multizone Buildings, *Energy and Buildings* **18:2**, 159-170.

Iman R.L., Davenport J.M., and Zeigler D.K. (1980). *Latin Hypercube Sampling (a Program User's Guide), Technical Report SAND79-1473*, Sandia Laboratories, Albuquerque NM, USA.

Sohn M.D., Daisey J.M., and Feustel H.E. (1998). Characterizing Indoor Airflow and Pollutant Transport Using Simulation Modeling for Prototypical Buildings. 1. Office Buildings. *Proceedings of the Eighth International Conference on Indoor Air Quality and Climate, Indoor Air 99*, Edinburgh, Scotland, **4**, 719-724.

Walton G.N. (1989). Airflow Network Models for Element-Based Building Airflow Modeling. *ASHRAE Transactions* **95:2**, 611-620.

Walton G.N. (1997). *CONTAM96 User Manual, Report NISTIR 6056*, U.S. Department of Commerce, National Institute of Standards and Technology, Gaithersburg MD, USA.

Wray C.P. and Yuill G.K. (1993). An Evaluation of Algorithms for Analyzing Smoke Control Systems. *ASHRAE Transactions* **99:1**, 160-174.