On the effects of decoupling air flow and heat balance in building simulation models

Per Sahlin[1], Ph.D.
Member ASHRAE

ABSTRACT

Several thermal building simulators allow also coupled modeling of bulk air movements using air flow network models. However, solving the combined flow and thermal problem can be problematic, both in the context of traditional building simulators and for modern environments, where both air flow and thermal models are formulated as sets of differential-algebraic equations (DAE). For DAE based simulators, difficult coupled problems often lead to small timesteps and excessive simulation times. In this paper, we will investigate the differences between full coupling, where all equations are solved simultaneously, and so-called "ping-pong coupling," where air-flows from the previous timestep are used. The latter approach is sometimes used in traditional simulators to simplify tool integration. Here, it will be investigated in the context of a variable timestep DAE sovler. The general motivation for DAE based building simulation is discussed and bi-directional air flow network models in this setting are presented in some detail. Results of the decoupling experiments are presented for a set of test cases.

INTRODUCTION

Buoyancy driven inter-zonal flows have a significant impact on the heat balances of the rooms also for airtight mechanically ventilated buildings. Airborne heat flux through the, usually open, internal doors of a building becomes significant even for moderate temperature differences between rooms. The closed doors and user-guessed air flows that are common practice in most multi-zone simulations are indeed a poor representation of reality. For this reason, the most advanced thermal building simulators have been extended to calculate also thermally and wind driven bulk air flows (EDSL 2002; SEL 2000; Hensen 1995). In most cases, algorithms and code from existing air flow network simulators such as COMIS (Dorer 2001) or CONTAM (Walton 1997) have been utilized for this co-simulation. Various closeness of coupling between the two categories of tools exist, carrying descriptions as "ping-pong" and "onions" "Ping-pong" coupling means that the airflow and thermal simulators are run in sequence, using air flows from the previous timestep as input for the thermal model. For "onions," the two models are iterated within each timestep until convergence is reached. For the "ping-pong" approach, there is a risk for instabilities if timesteps become large. In the "onions" case, the combined model should behave physically correct once a solution is obtained, but there may be difficulties in obtaining convergence within each timestep.

An alternative to co-simulation is given by new simulation environments that base their models on a set of differential and algebraic equations (DAEs). Here, the thermal and air flow equations are on equal footing and, being part of the same system of equations, they are normally solved simultaneously achieving full coupling between the airflow and thermal parts of the model. From a dynamical stability point of view, this is equivalent to "onions" co-simulation.

Modern DAE based simulation environments are equipped with variable timestep solvers, which adapt the integration timestep continuously while maintaining control of solution accuracy. Experience shows that in some cases with natural ventilation time steps may become quite small, resulting in long execution times.

Based on an equation based representation of the air flow network and thermal model, this work seeks to investigate the differences between fully coupled and "ping-pong" coupling. We will study whether the

[1] Per Sahlin is a partner and CEO of Equa Simulation AB, www.equa.se, Stockholm, Sweden

smaller system of equations obtained for the "ping-pong" case will be easier to solve with comparatively long steps – however short enough to be stable from the airflow-thermal coupling point of view.

In the next section, some background and motivation will be given about DAE based simulation environments and associated modeling languages. The model library used for this study, IEA Task 22 Library, will be briefly introduced and the airflow network models of the library will be more thoroughly discussed. In the subsequent section, the experiment will be described, starting with the decoupling of airflow and thermal equations, continuing with a description of the various test cases and ending with a discussion of accuracy of the DAE integration process. The results of the investigation are presented in the fourth section, followed by discussion and conclusions in the final sections.

DAE BASED SIMULATION ENVIRONMENTS

Today engineers and scientists in most industrial sectors spend significant efforts on the development of mathematical models and associated simulators enabling virtual experiments. Traditionally, the mathematical models have been developed in close conjunction with the programming of solution algorithms and other parts of the simulation program, eventually resulting in large "codes" enabling study of the target engineering domain. The well-known whole-building simulators such as DOE-2, BLAST and EnergyPlus represent examples of such monolithic simulation codes, containing many years of engineering development effort. Parallel examples exist in most engineering disciplines.

A major problem with virtually all traditional simulators is that, in time, their complexity tends to become quite overwhelming, making it difficult – also for the original developers – to make changes and additions as user needs evolve. System architecture and mathematical solution methods then become issues and the need may even arise to re-write major parts to attain a better, more manageable, basic structure.

In the development process of - for example - a modern automobile, several complex stand-alone hand-coded simulators are utilized for various physical processes such as engine combustion, engine motion, drive train, brake system, whole-car dynamic motion etc. Each of these simulators have been developed to depict all essential phenomena involved in the subsystem covered. However, to take the next step of virtual experimentation in the development process, there is a need to couple the stand-alone simulators, approaching a whole-car simulation model. However, experiments with so called co-simulation involving two or more existing domain simulators have not generally been successful and developers have begun to look for an entirely new approach (Tiller et al. 2000).

A natural solution to the problem of simulation software complexity is to separate the simulation models (the mathematical system descriptions) from the tools, most importantly the mathematical solvers, but also model manipulators, result presentation tools etc. Let us call all these tools collectively the *simulation environment*. By this separation, several independent developers of domain specific *model libraries* may concentrate on their domain knowledge, leaving the development of the simulation tools to other experts. Furthermore - and this is in most cases the main argument - multiple domain-specific models may be coupled without need for co-simulation. Given the advantages, a large number of such domain independent simulation environments have been developed and reached proliferation ever since the early sixties and most low- to moderate-complexity simulation models indeed reside in such a setting. However, until recently, almost without exception, the most involved engineering models have been developed without such tools.

In 2000, at Ford Motor Company, a serious attempt to try state-state-of-the art DAE based simulation methods on the whole-car problem resulted in the development of a large equation model, with full complexity in each subdomain. The model proved to be sufficiently robust and efficient and this experiment has resulted in rapidly growing application of DAE methodology in the automotive industry (Bowles 2001).

If all the relationships that govern a thermal building simulation problem are written down as equations (partial differential equations are first discretized in space,) one arrives at a so called hybrid DAE problem. "DAE" stands for differential-algebraic equations, i.e. a mixture of ordinary differential and algebraic equations. "Hybrid" denotes the fact that the system may contain discontinuities and hysteresis. The resulting system of equations will contain many different time-scales and have several non-linear equations. Corresponding system matrices will have no special properties, such as being symmetric, diagonally dominant or positive definite. In the equation based method, this complex system of equations is formulated and solved using general-purpose methods, i.e. methods that have no strong relationship to any engineering field or more specific problem category. Hybrid DAE:s occur naturally in many domains such as chemical process engineering, analog electronics, mechatronics, multi-body mechanics, power

generation and distribution, electrical power and pipe flow networks to name a few. Some commercial software systems that deal with this class of problems have been presented by Jeandel et al. (1997), Elmqvist et al. (2001), Stangerup (1991), Oh and Pantelides (1996) and Equa (2001). In addition to these, there is a large number of academic systems: ASCEND (Piela et al. 1991), OmSim (Andersson 1994), SMILE (Jochum 1993) and SPARK (Buhl et al. 1993) to name a few.

The DAE problems are formulated using special *modeling languages*. Several such languages have been developed, the first being Dymola (Elmqvist 1978) which was defined in 1978. NMF (the Neutral Model Format) was proposed as a possible standard for building simulation models in 1989 (Sahlin and Sowell 1989). The first formal standard is VHDL-AMS (IEEE 1997), an extension of the VHDL language for logical circuitry simulation. Modelica (www.modelica.org) was proposed in 1997 as the first completely domain-independent standard.

We will not digress here on the issue of DAE modeling language design - it is a research area of its own - but merely point out a few main characteristics about this way of expressing models. The two main advantages of using such a language are:

- The modeler can forget entirely about the way equations are solved, he/she is merely required to *formulate* the model, not to *solve* it. This enables domain experts to concentrate on their specialty and not have double duty as numerical analysts.
- The resulting model can be automatically transformed in many ways. Most importantly, it can be translated into the format required by more than a single simulation environment. Most often, generated code is in the form of directly executable C or Fortran. But it is also possible to translate into a different DAE modeling language and continue to work with the code as source in the new language. This means that available model material is not tied to a single language as long as the required translators are available.

Translation from a DAE language to algorithmic, executable code can thus be done. Indeed, this is the main feature of the DAE based systems. However, automatic translation of existing C or Fortran into a DAE language is not possible. Nevertheless, it is often possible to re-use algorithmic code in the DAE context by calling existing routines from the DAE model.

EQUATION MODEL

IEA Task 22 Library

Within a recent project of the International Energy Agency, SH&C Task 22 Subtask B, a significant attempt was made to develop a comprehensive DAE based library of building simulation models (Bring et al. 1999; Vuolle et al. 1999). At the core of this library are a detailed and a simplified zone model. The detailed zone model with full Stefan-Boltzman long-wave radiation has been developed primarily for indoor climate studies and design tasks. With this model, it is possible to study displacement ventilation, mean radiant and operative temperatures, comfort indices and daylighting. The simplified zone model has been developed for energy simulations. Both models have balance equations for CO_2, humidity and energy. The CO_2, moisture and heat loads from people are modeled according to ISO/DIS 7730 (ISO 1984).

A key feature of the library is the modeling of thermal as well as air flow problems, the aspect that will be further investigated in this paper.

The library also has component models for primary and secondary HVAC systems. These models are designed to have a minimum number of supplied parameters and include ideal equipment control. For more detailed secondary system simulations, the ASHRAE Secondary Toolkit models have been translated into NMF, and they are compatible with the other models of the library.

Since the conclusion of the Subtask B project, the IEA Task 22 Library has been significantly improved and the commercial building simulator used for this study, which is entirely based on the library, has been successfully introduced on the market (Equa 2002).

A thorough account of the IEA Task 22 Library is beyond the scope of this paper and we will instead in the next sections focus on the specific issue of bidirectional airflow network modeling in the context of an NMF based DAE library.

Air Flow Network Models

In an air flow network library there are two basic groups of components; nodes and connecting elements. The node components are characterized by their potentials, e.g. pressure (P) and temperature (T). Nodes can be zones, or junctions in the ventilation system. Their model equations represent conservation of mass and energy:

$$0 = \sum m_i \qquad (1)$$

$$0 = \sum q_i + q_{zone} \qquad (2)$$

where q_{zone} is a heat source, possibly from energy transfer with the envelope.

The connecting elements can be leaks, or any double-ended pieces of the ductwork (ducts, grilles, fans, etc.). Among the model equations of a connecting element there is always the mass flow modeled as a function of the pressure difference over the element:

$$m_{1\text{-}2} = f(P_1 - P_2) \qquad (3)$$

In the IEA Task 22 Library, power law equations are used for the pressure - flow relation. For a leak this implies:

$$f(\Delta p) = \begin{cases} c(\Delta p)^n, & \Delta p > 0 \\ -c(-\Delta p)^n, & \Delta p < 0 \end{cases} \qquad (4)$$

where n is a coefficient between 0.5 and 1.0, depending on flow type, and c is a "conductivity" coefficient.

Heat transport through a connecting element is for a dry-air case:

$$q_{1\text{-}2} = \begin{cases} c_p T_1 m_{1\text{-}2}, & m_{1\text{-}2} > 0 \\ c_p T_2 m_{1\text{-}2}, & m_{1\text{-}2} < 0 \end{cases} \qquad (5)$$

where T_1 and T_2 are the temperatures of the nodes connected by the element.

Equations (1) – (5) and corresponding NMF code below are a simplified versions of code from the IEA Task 22 Library. The full version of the library with and without the decoupling done for this study can be found at http://www.equaonline.com/air_decoupling .

NMF Examples

In order to illustrate some basic bidirectional air flow modeling in NMF, two examples are shown below. The NMF code in the first example models a zone with three airflow connection points, called `LINKS`, and the next a leak between zones.

The link type in the example, `BidirAir`, handles air-exchange interaction in terms of bi-directional flow between neighboring models. It has four variables:

```
BidirAir       (Pressure, MassFlow, Temp, HeatFlux)
```

The zone model also has a `Tq` link. It is an interface for temperature and heat flow between the airflow family of models and a thermal model for the building envelope.

```
CONTINUOUS_MODEL  simple_bdzone

ABSTRACT
  "A static zone model for airflow modeling. Bidirectional
   transports of energy is modeled."

EQUATIONS

/* mass conservation (eqn (1))*/

   0 = M_0 + M_1 + M_2;

/* energy conservation (eqn (2))*/

   0 = Q_zone + Q_0 + Q_1 + Q_2;

LINKS

  /* type     name        variables... */

BidirAir    terminal_0    P, POS_IN M_0, T, POS_IN Q_0;
BidirAir    terminal_1    P, POS_IN M_1, T, POS_IN Q_1;
BidirAir    terminal_2    P, POS_IN M_2, T, POS_IN Q_2;

Tq          air_temp      T, POS_IN Q_zone;

VARIABLES

  /* type      name   role [def   min  max]  description */

  MassFlow     M_0    OUT               "terminal 0 massflow"
  MassFlow     M_1    IN                "terminal 1 massflow"
  MassFlow     M_2    IN                "terminal 2 massflow"
  Pressure     P      IN                "zone pressure"
  HeatFlux     Q_0    OUT               "terminal 0 HeatFlux"
  HeatFlux     Q_1    IN                "terminal 1 HeatFlux"
  HeatFlux     Q_2    IN                "terminal 2 HeatFlux"
  Temp         T      IN                "zone temperature"
  HeatFlux     Q_zone IN                "heat gain/loss in zone"

END_MODEL
```

The zone has a pressure and a temperature that is implicitly defined by neighboring components. These variables do not occur explicitly in the zone equations, but only on the links. The `role` attribute of the variable declarations are used to generate code for input-output oriented simulation environments such as TRNSYS and HVACSIM+.

```
CONTINUOUS_MODEL simple_bdleak

ABSTRACT "A simplified powerlaw leak model w/ bidirectional transports tempered air."

EQUATIONS

/*driving pressure difference*/

  Dp = P1 - P2;

/* powerlaw massflow equation (eqns (3) and (4))*/

  M = IF Dp > 0 THEN c * Dp**n
      ELSE -c * (-Dp)**n
      END_IF ;

/* convected heat through leak (eqn (5))*/

  Q = IF M > 0.0 THEN cp * T1 * M
      ELSE cp * T2 * M
      END_IF ;

LINKS

/* type      name        variables... */

BidirAir    terminal_1   P1, POS_IN  M, T1, POS_IN  Q
BidirAir    terminal_2   P2, POS_OUT M, T2, POS_OUT Q;


VARIABLES

/* type    name  role def  min  max  description */

MassFlow     M    OUT  0   -BIG    BIG  "massflow through leak"
Pressure     P1   IN   1   -BIG    BIG  "terminal 1 pressure"
Pressure     P2   IN   2   -BIG    BIG  "terminal 2 pressure"
Temp         T1   IN   20 ABS_ZERO BIG  "Temperature of neighbor 1"
Temp         T2   IN   20 ABS_ZERO BIG  "Temperature of neighbor 2"
HeatFlux     Q    OUT  0   -BIG    BIG  "heat moved by massflow"
Pressure     Dp   OUT                   "effective pressure diff."

PARAMETERS

/* type   name  role def  min  max  description */
Generic   c     S_P  1    0     BIG "powerlaw coeff. [kg/(s Pa**n)]"
Generic   n     S_P .5   .5     1.0 "powerlaw exponent [dimless]"
HeatCapM  cp    S_P  1006 .5E3  3E3 "air cp"
END_MODEL
```

These two models can be combined into arbitrary networks. The connection rules are that two zones always must be connected via one or more leaks. Leaks may in principle be connected in series.

Key to the understanding of these models is the fact that the pressure and temperature that appear in the leak model are valid for the *neighboring* components. `P` and `T` of the zone model become computable since they occur in equations in the leak model.

Some Real-Life Issues

The airflow network models are highly non-linear and therefore rather demanding to solve numerically. For non-linear problems a solver must start with an initial guess of the solution. For general-purpose solvers the initial guess is normally supplied by the user. Given this guess, the solver will try to improve the solution using various schemes. The quality of such numerical schemes is determined by their ability to safely and quickly produce a solution given a poor initial guess.

The simple models that we have just discussed, will work well for good quality initial guesses, if the massflows in the solution process stay away from zero. One problem is that the derivative of Equation (4) with respect to Δp becomes infinite around zero, and since the purpose of the models is to predict bi-directional flow, they should be robust for evaluation in the neighborhood of zero massflow.

Solution ideas to these types of difficulties can often be obtained from the physics of the problem. In this case, for a small enough flow through an opening, there will be a switch to laminar flow, with a corresponding linear relationship between flow and pressure. If Equation (4) is augmented with a linear section around zero, we should be able to avoid the problem:

$$f(\Delta p) = \begin{cases} c(\Delta p)^n, & \Delta p > \Delta p_0 \\ c_0 \Delta p, & \text{abs}(\Delta p) < \Delta p_0 \\ -c(-\Delta p)^n, & \Delta p < -\Delta p_0 \end{cases} \quad (4b)$$

where the parameter Δp_0 could be selected to be smaller than any interesting pressure difference to resolve. c_0 can be computed to give a continuous transition between the different regimes of Equation (4b).

Another problem with the air flow models, as with any non-linear model, is that they require a reasonable initial guess to be provided in order to converge safely.

NMF provides a way around this problem by a system function called `LINEARIZE`. If a solver supports this feature, a call to `LINEARIZE` will return `true` in the first few iterations. The model can then detect this and behave linearly or close to linearly in this case, and thus provide a user-independent and reasonable initial state, since most solvers will solve a linear system in one iteration independent of the starting point.

Equations (3) - (5) in the leak model, with both isolation of the singularity around zero and a call to `LINEARIZE`, look like:

```
/* powerlaw massflow equation */

  M =       IF LINEARIZE (1) THEN c * Dp
            ELSE_IF abs (Dp) < dp0 THEN c0 * Dp
            ELSE_IF Dp > 0 THEN c * Dp**n
            ELSE -c * (-Dp)**n
            END_IF ;

/* convected heat through leak*/

  Q = IF LINEARIZE (1) THEN (T1 - T2) / 2
      ELSE_IF M > 0.0 THEN cp * T1 * M
      ELSE cp * T2 * M
      END_IF ;
```

The linearized model for the heatflux Q deserves some comment. It makes no effort to depict reality. However, if similar constructs are used in other components, the temperatures reached in the linear stage will be averages between boundary temperatures. Given the weak coupling from temperature to pressure in normal ventilation studies, this should mean realistic temperatures in most cases. Thus, this model will not aggravate the difficulties with the non-linear flow balance.

In the previous section, excerpts and simplified versions of the real air flow models of the Task 22 librarys have been presented. The full library also model some additional physics that go beyond the scope of this paper:

Stack effect	Driving forces due to temperature dependent density change are modeled throughout the library.
Contaminant fraction	In addition to the transport of heat with the air, a fraction of some arbitrary substance is also modeled throughout the circuit. The standard zone models of the library assumes the fraction to be CO_2.
Moist air	The full library takes account of air moisture storage and transport.

Model for Vertical Temperature Gradient

The detailed zone model of the IEA Task 22 Library includes a simple model for calculation of a vertical temperature gradient by Mundt (1996). The model has been validated primarily for displacement ventilation of office spaces but, lacking better practical alternatives, it has been used for many other cases such as glazed atria, historical buildings and seems to give reasonable agreement with measurements also for these cases.

The zone air volume is described by an air node with heat capacity. The temperature of this node is valid at the ceiling of the zone. All heat transfer to the zone air is fed into this node. Equation (6) is used to calculate an air temperature at floor level and temperatures at zone surfaces and air terminals are interpolated between air temperatures at floor and ceiling levels.

The air temperature at floor level is calculated from a heat balance between supply air flow and convection at the floor

$$\dot{m}_{air}\ c_{pair}(T_{AirFloor} - T_{Supply}) = h_c\ A\ (T_{Floor} - T_{AirFloor}) \quad (6)$$

where

m_{air} is supply air massflow, kg /s
c_{pair} is air heat capacity, J/kgK
$T_{AirFloor}$ is air temperature above floor, °C
T_{Supply} is supply air temperature, °C
h_c is a convective heat transfer coefficient, W / K m^2
A is floor area, m^2
T_{Floor} is floor temperature, °C

If the calculated floor air temperature is lower than the air temperature at the ceiling, the temperature gradient is accepted. On the other hand, if the calculated temperature gradient would be negative, a well-mixed model is used instead and all air temperatures are set equal.

The NMF code for the detailed zone model (`ceDetZon`) which includes the gradient model can be found on http://www.equaonline.com/air_decoupling, where also the model for large vertical openings (`ceLVO`) can be studied. The latter is derived directly from Bernoulli's equation with a correction for viscous effects.

THE EXPERIMENT

Decoupling Airflow

For the moist air case, Equation (5) for energy transport through a connecting element becomes

$$q_{1\text{-}2} = \begin{cases} m_{1\text{-}2}Enthal(T_1, H_1), & m_{1\text{-}2} > 0 \\ m_{1\text{-}2}Enthal(T_2, H_2), & m_{1\text{-}2} < 0 \end{cases} \quad (5mo)$$

where $Enthal(T_1, H_1)$ and $Enthal(T_2, H_2)$, are the enthalpies of the nodes connected by the element. In the standard version of IEA Task 22 Library (fully coupled), the massflows are valid for the same point of time as all other quantities.

For the decoupled case, Equation (5mo) becomes

$$q_{1\text{-}2} = \begin{cases} m_{1\text{-}2,old}Enthal(T_1, H_1), & m_{1\text{-}2,old} > 0 \\ m_{1\text{-}2,old}Enthal(T_2, H_2), & m_{1\text{-}2,old} < 0 \end{cases} \quad (5de)$$

where the subscript *old* denotes massflows from the previous timestep.
Old massflows have similarly been used in the transport equations for humidity and CO_2.

The decoupling has been introduced in all instances of such transport equations. The following models of IEA Task 22 Library were altered:

`ceLeak`	the model for leaks and small openings
`ceLVO`	the model for large vertical openings
`ceSupT`	the model for mechanical supply air flow control (and supply terminal)
`ceExhT`	the model for mechanical exhaust air flow control (and exhaust terminal)

Testcases

The following cases have been tried:

- a family apartment

- a two-zone CAV office with an open door
- a temperature controlled VAV ditto
- a single-zone office with natural ventilation

The last case replaced a single family, naturally ventilated house with a two-story living room with a significant vertical temperature gradient. This case failed for all parameter selections using the decoupled model, making it less interesting for any detailed presentation.

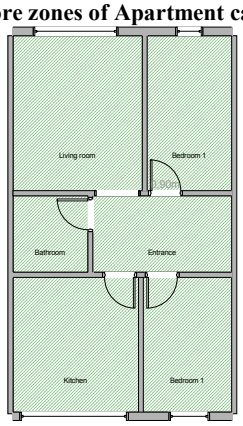
Apartment. The model was developed by Bengt Bergsten, CIT Energy Management AB, for a large simulation study of natural ventilation and heat exchange between apartments in some common Swedish building types. For the sake of this paper, a January week is simulated.

Six core zones (Figure 1) plus five zones to describe conditions in surrounding spaces (a total of 11 Detailed Zone models). Five internal doors with opening schedules. Nine windows, four of which have opening schedules. 17 leaks. Nine water radiators with proportional controllers. 26 (detailed) schedules, controlling openings, occupant presence, lights and other loads.

Chiller (unused) and boiler. Core zones have a CAV mechanical exhaust ventilation system, 0.35 l/s m^2, unconditioned supply through window ventilators.

9898 variables in 316 NMF component instances.

FIGURE 1
Core zones of Apartment case

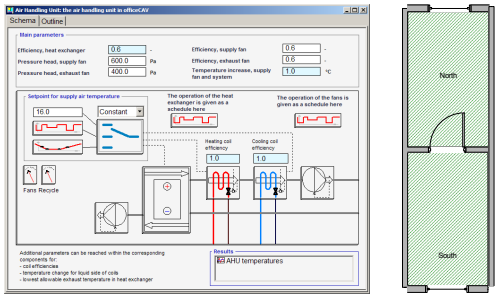


CAV Office. Developed as a simple test case for this paper. For the timing experiments, the month of July is simulated. Two well-mixed (detailed) zones, a loaded south zone and an adjacent unloaded north zone, joined by an open common door. Two permanently closed windows. Two leaks towards ambient. All loads (in South zone) always on.

Boiler (unused) and chiller. Balanced mechanical CAV system, 2 l /s m^2. Supply air, permanently on, kept at 17°C by an air-to-air heat exchanger followed by heating and cooling coils.

2057 variables in 69 NMF component instances.

FIGURE 2
AHU Model and Floor Plan of Office case



VAV Office. Same as CAV office but with a temperature controlled VAV system. Ventilation air is increased proportionately from 0.3 to 7 l /s m^2 as room air temperature increases from 21 to 23 °C.

2067 variables in 70 NMF component instances.

Gradient Office. Developed for this paper to test some case with a vertical temperature gradient. Same properties as loaded (South) zone in CAV Office, but with calculation of a vertical temperature gradient and two large leaks in the external wall 1.5 vertical meters apart. The Equivalent Leakage Area of each leak (at 4 Pa) is 0.10 m^2 to get a significant natural ventilation contribution to the heat balance.

1365 variables in 50 NMF component instances.

Location and Climate Data

The location of all cases was selected to be Copenhagen, Denmark, a Test Reference Year Climate file was used for weather data.

All details about the test cases have been made available at http://www.equaonline.com/air_decoupling. Other researchers are encouraged to investigate the cases in other simulators.

DAE Solver Issues

A commercial building simulator based on the (public domain) IEA Task 22 Library was used for the experiments (Equa 2002). The building simulator has been developed in a general-purpose DAE-based simulation environment which relies on model descriptions in NMF or Modelica (Equa 2001).

The native DAE solver of the simulation environment was used for the experiments. The solver has been designed to be extremely robust and offers a range of numerical methods. For implicit integration methods, the solver can - continuously during the simulation - adapt the order (number of previous timesteps that are used) and, more importantly, the integration timestep. For the current experiments, BDF1 (The first order backward Euler method) was used and thus automatic order selection was not utilized.

The accuracy of the integration is controlled in each timestep by comparing the calculated solution with the predicted solution. Deviations are checked at the variable level as prescribed by solver parameters *tol* and *tol_lim*. Deviations are calculated relative to variable values, if these are larger than *tol_lim*, otherwise relative to *tol_lim*. If the accuracy is insufficient, i.e

$$|dy|/|y| > tol, \qquad |y| > tol\_lim$$

$$|dy|/tol\_lim > tol, \qquad |y| \le tol\_lim$$

the step is retraced and a shorter step is tried.

.

During normal timesteps, the solution is calculated by Newton-Raphson iterations, which is continued until the largest change between iterations satisfies the criterion

$$|dy|/|y| \le tol * newton\_tol \quad , \qquad |y| > tol\_lim$$

$$|dy|/tol\_lim \le tol * newton\_tol, \qquad |y| \le tol\_lim$$

where the solver parameter *newton_tol* controls the accuracy of the iteration process.

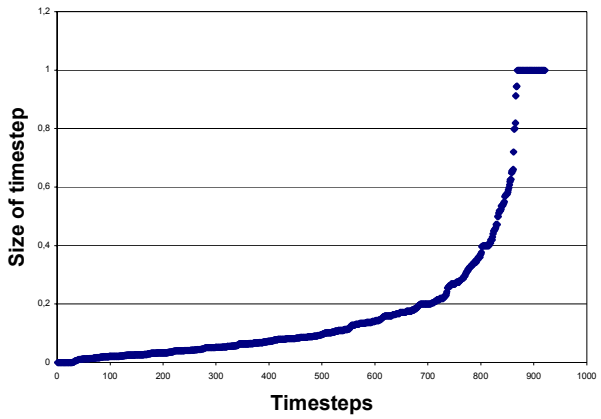
For the study, *tol_lim* was maintained at 10 and *newton_tol* was varied to keep *tol* * *newton_tol* constant at 0.001.

A solver parameter *h_max* (in the experiment with the unit hours) allows the specification of a maximum timestep to be taken. Normally, the timestep should be controlled by tolerances. However, specifying some bound on the allowed timestep avoids problems with excessively long steps that may, for example, completely overshoot small features of input time-series.

To mimic the behavior of a fixed timestep environment, some experiments were carried out with an extremely loose tolerance parameter (*tol* = 1), leaving the size of the maximum timestep largely to be governed by *h_max*. However, after discrete events (signaled discontinuities in equations or driving time-series), much smaller timesteps will be taken directly after the event, creating a continuum of timesteps from very small to h_max. Also if the sheer non-linearity of the equations create convergence problems for undamped Newton-Raphson at large timesteps, the solver will decrease the timestep.

In the apartment case, the non-linearities were such that large timesteps were rarely taken. Figure 3 shows a sample timestep distribution for a case with *tol* = 1 and *h_max* = 1.

FIGURE 3
Distribution of timesteps for Apartment Case with *tol* = 1 and *h_max* = 1



All experiments were carried out on a 1.6 MHz Pentium 4 laptop.

RESULTS

Table 1 shows the results for the Apartment case using full coupling for a variety of tolerances and maximum timesteps. *sim. time* is the measured execution time, *no. steps* the total number of successful timesteps taken and *p_max* is the maximum heating power supplied during the simulated January week. The latter is presented as an indicator to provide some physical estimate of solution accuracy. For the Apartment case, the run with *h_max* = 0.01 can be regarded as a "truth case" for the indicator variable. For the other cases, the fully coupled variant with the finest tolerance should be the most reliable.

TABLE 1: Apartment case – fully coupled

tol	h_max	sim. time [s]	no. steps	p_max [W]
0.002	1.5	88.0	3 424	10 888
0.005	1.5	71.2	2 524	10 884
0.01	1.5	66.1	2 130	10 884
0.02[2]	1.5[1]	63.1	1 891	10 886

[2] The default settings for the building simulator

1	0.01	324.6	17 318	10 878
1	0.02	184.5	9 074	10 890
1	1.0	61.6	1 541	10 884
1	2.0	60.7	1 523	10 871

All combinations of parameters seem to yield a fairly good accuracy for *p_max*. Most likely this is because the timesteps remain quite small as previously discussed also for large *h_max*. The average timestep for the last case is for example 0.11 h.

Table 2 shows the corresponding results with the decoupled model. When timesteps are governed by the *tol* parameter, there is indeed a small gain in execution time. However, for any case where some longer timesteps are attempted, the simulation failed completely. In order to get a solution for a "fixed" timestep case (*tol* = 1), an extremely small *h_max* had to be selected.

TABLE 2: Apartment case – decoupled

tol	h_max	sim. time [s]	no. steps	p_max [W]
0.002	1.5	76.6	3 344	10 886
0.005	1.5	60.0	2 387	10 885
0.01	1.5	failed		
0.02	1.5	failed		
1	0.01	298.7	17 249	10 878
1	0.02	failed		
1	1.0	failed		
1	2.0	failed		

Table 3 shows the fully coupled CAV Office case. This is a free floating case and no peak power is available as indicator. The maximum of the operative temperature in the middle of the floor of the loaded south room has been selected as indicator. Only small differences in the indicator can be discerned between the runs.

TABLE 3: CAV Office case – fully coupled

tol	h_max [h]	sim. time [s]	no. steps	t_op_s_max [°C]
0.002	1.5	12.0	1 941	27.59
0.02	1.5	8.9	1 294	27.53
0.2	1.5	6.3	981	27.55
1	0.5	10.0	1 868	27.58
1	1.0	6.4	1 114	27.55
1	2.0	5.0	778	27.64

Table 4 represents the decoupled CAV Office case. For this case, there are no significant differences in either results or simulation performance measures.

TABLE 4: CAV Office case – decoupled

tol	h_max [h]	sim. time [s]	no. steps	t_op_s_max [°C]
0.002	1.5	12.1	1 943	27.58
0.02	1.5	7.7	1 253	27.50
0.2	1.5	6.0	980	27.55
1	0.5	10.7	1 868	27.58
1	1.0	7.1	1 115	27.55
1	2.0	5.6	816	27.53

Table 5 shows the same case but with a temperature controlled VAV system. Maximum cooling coil power has been selected as indicator. Here there are indeed some variations in accuracy depending on the solver parameters selected. The peak of the cooling coil power will be well captured if there happens to be a timestep near it, which is more likely to occur for small steps.

TABLE 5: VAV Office case – fully coupled

tol	h_max [h]	sim. time [s]	no. steps	p_max [W]
0.002	1.5	14.1	2 207	2 092
0.02	1.5	9.3	1 457	1 946
0.2	1.5	7.3	1 111	2 078
1	0.5	10.6	1 945	2 088
1	1.0	7.8	1 286	1 996
1	2.0	6.2	935	2 059

In Table 6, the decoupling clearly shows an adverse effect on both simulation performance and predictive accuracy for the VAV Office. The coil power is fluctuating for reasons of numerical instability (we will show an example of this later), resulting in an either over- or under-predicted *p_max* whenever larger timesteps are attempted.

TABLE 6: VAV Office case – decoupled

tol	h_max [h]	sim. time [s]	no. steps	p_max [W]
0.002	1.5	22.2	3 677	2 102
0.02	1.5	16.2	2 415	2 210
0.2	1.5	13.6	1 855	2 166
1	0.5	20.5	2 792	2 313
1	1.0	15.3	1 977	2 046
1	2.0	14.1	1 756	2 337

The Gradient Office case (the fully coupled variant shown in Table 7), is free floating and the maximum air temperature just below the ceiling has been selected as indicator. Only insignificant variations in the indicator is found in the fully coupled case.

TABLE 7: Gradient Office case – fully coupled

tol	h_max [h]	sim. time [s]	no. steps	t_ceil_max [°C]
0.002	1.5	10.6	2 382	29.22
0.02	1.5	5.9	1 299	29.16
0.2	1.5	4.6	960	29.20
1	0.5	7.6	1 853	29.22
1	1.0	4.8	1 085	29.21
1	2.0	3.7	767	29.16

In Table 8, it is clear that for the decoupled Gradient Office case, the decoupling again has a negative effect on both accuracy and numerical performance. Unacceptable deviations in the indicator variable are shown also for small timesteps.

TABLE 8: Gradient Office case – decoupled

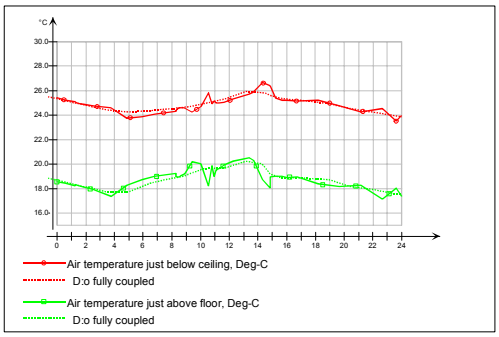
tol	h_max [h]	sim. time [s]	no. steps	t_ceil_max [°C]
0.002	1.5	18.8	3 976	29.35
0.02	1.5	9.4	1 937	29.94
0.2	1.5	6.3	1 201	29.71
1	0.5	8.6	1 890	29.96
1	1.0	6.7	1 295	29.66
1	2.0	6.2	1 079	31.28

DISCUSSION

Slight runtime gains can be found in the Apartment and CAV Office cases. However, these are far outweighed by the adverse effects shown by the decoupling. For the Apartment case, several runs even failed completely; the decoupling experiment must be regarded as a failure from the perspective of improving the performance of IEA Task 22 Library in context of the DAE solver used. Figure 4 shows the floor and ceiling air temperatures for the last day of simulation with the Gradient Office case. The decoupled version shows clear numerical instabilities, which will result in both lower predictive accuracy

and in longer simulation times, since the solver will have to deal with the instabilities by reducing the timestep.

FIGURE 4
Ceiling and floor air temperatures for the Gradient case during a 24 h period.



Alternative methods to save DAE integration time

The problem with too small steps for the Apartment case – and similar cases – remains. From a viewpoint of needed engineering accuracy, the DAE model and solver combination is simply too accurate; using the standard tolerance controls it seems impossible to take sufficiently large steps. It was decided to make some additional exploratory experiments with the Apartment case, to see what could be done if the model was somewhat reformulated. The results are summarized in Table 9. Another indicator variable has been added, the total of purchased (heating) energy, since for many real simulation situations this will be the most important result.

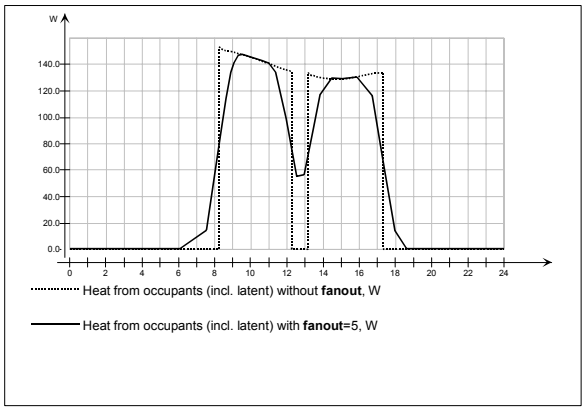
TABLE 9: Apartment case – speed optimization

fanout=5	simpl. zone	kmax=2, n_jac=4, tol=0.2	no internal LVOs	sim. time [s]	no. steps	p_max [W]	purchased energy [kWh]
				63.1	1 891	10 886	1 030
x				34.1	894	10 124	1 027
x	x			25.9	950	10 001	1 013
x	x	x		20.6	684	9 921	1 015
x	x	x	x	13.2	480	9 946	1 015
x		x	x	17.2	454	10 037	1 035

The first row of data shows the starting point, using default solver parameters. The first measure is called *fanout*, which is a scheme to automatically smoothen sharp features of driving schedules, which in this case are used for load profiles and window and door opening operations. The dotted line in Figure 5 shows an occupant load as the operative temperature increases from 21.2 to 23.8 [°C], entry at 8.15, 56 minute lunch at 12.15, exit at 17.20. The result after smoothing is shown with the solid line.

Smoothing of schedules may naturally affect some simulation results. For the Apartment case, this is indeed the situation with the peak heat load while the energy consumption remains relatively unaffected. For this case, smoothing nearly cuts the simulation time in half by eliminating several event calls that upset the sequence of timesteps. Each event is passed by special solution techniques and after the passage the solver starts with a small step and increases it successively.

FIGURE 5
An occupant load with and without smoothing.



In a second run, the Detailed zone model is replaced by the *Simplified zone* model. This reduces the total number of variables from 9898 to 8832 (11 %). Accuracies for both indicator variables are affected by a few percent and simulation time is reduced by another 13 %. The simplified zone model uses a mean radiant temperature strategy and should normally be sufficient for this case.

In the third experiment, some other *solver parameters* are changed. The maximum order of the integration method is allowed to go up to a second order method (*kmax* = 2), Jacobians are recalculated every fourth timestep unless there are convergence problems (*n_jac* =4, default is 2) and a looser overall tolerance is used (*tol* = 0.2). This saves yet another eight percent of execution time at the expense of another percent additional loss of accuracy in the peak power measure.

In the fourth run, the actual model is altered. The modeler had specified five *internal doors*, two of which had opening schedules. It was not immediately obvious that modeling of internal bidirectional air-flows through the internal doors would have a significant effect on the overall result, since zone radiators had the same setpoints in all rooms and ventilation air would govern the flow direction. Replacing the internal door models with large-aperture leak models, which only allow airflow in one direction at the time, indeed saved yet another 12 %, again with small losses in accuracy. All measures combined provided a 79 % reduction in execution time.

In a final experiment, the detailed zone model was reapplied, resulting in fewer timesteps, some increase in execution time but also in somewhat better accuracy. In the final run, the average timestep is 0.37 h, approaching “normal” energy simulation timesteps.

DAE Based Building Simulators

Given a powerful user interface, with objects like openable doors and windows, chimneys etc., most users will naturally build models that present real challenges to a general-purpose DAE solver by containing severe non-linearities in combination with flow coefficients (Jacobian elements) differing by several orders of magnitude. The default “reaction” by the DAE solver is then to decrease the timestep to minimize the problems encountered and to maintain the required mathematical accuracy. This leads to timesteps that are perceived as being too small from a physical, engineering perspective.

As we have shown in the previous section, a range of remedies can be found to get the mathematical and engineering accuracies synchronized again. However, automating the process of tuning the model and solver parameters, making it practically accessible also to inexperienced users, is a challenging task. The building simulator used was first released in 1998 and significant improvements have been made in the version used here but there is clearly more to be done.

The advantages of applying rapidly evolving general-purpose DAE tools to building simulation problems are rather obvious and have been discussed in a previous section of this paper. The existence of a commercially appreciated[3] building simulator proves the concept. However, the mainstream building simulators have been tuned against user-needs for decades and will not be immediately replaced.

CONCLUSIONS

[3] The building simulator of this study has more than a thousand registered users, mainly in Scandinavia

- The “ping-pong” decoupling provided minor improvements in computational efficiency for some cases.
- It also gave rise to fatal instabilities, making the whole approach utterly unattractive as a means of improving numerical performance. In fact, “ping-pong” decoupling is most likely not working well in any setting where it is applied.
- A combination of other means proved effective in dealing with the problematic Apartment case and packaging and automating the application of such means seems to be a viable alternative.

REFERENCES

Andersson M. 1994. Object-Oriented Modeling and Simulation of Hybrid Systems. PhD thesis ISRN LUTFD2/TFRT--1043--SE, Department of Automatic Control, Lund Institute of Technology, Lund, Sweden, December 1994

Andre, Ph. ; Kummert, M. ; Nicolas, J. 1998. Coupling thermal simulation and airflow calculation for a better evaluation of natural ventilation strategies. Proc. Systems Simulation in Buildings ‘98, Liège, December 1998

Bowles, P. 2001. Feasibility of Detailed Vehicle Modeling. SAE-2001-01-0334. Society of Automotive Engineers.

Bring, A., P. Sahlin and M. Vuolle, Models for Building Indoor Climate and Energy Simulation, Report of IEA SHC Task 22 Building Energy Analysis Tools Subtask B Model Documentation, KTH, Stockholm, Sweden, 1999 (www.equa.se/dncenter/T22Brep.pdf)

Dorer, V. A. Haas , W. Keilholz, R. Pelletret, A. Weber 2001. COMIS v3.1 simulation environment for multizone air flow and pollutant transport modelling, Proc. of the IBPSA Building Simulation '01 conference, Rio de Janeiro, Brazil, 2001

EDSL 2002. A-Tas Theory Manual. Environmental Design Solutions Ltd.

Elmqvist H. 1978 A Structured Model Language for Large Continuous Systems. PhD thesis, ISRN LUTFD2/TFRT--1015--SE, Department of Automatic Control, Lund Institute of Technology, Lund, Sweden, May 1978

Elmqvist H., D. Brück, and M. Otter. 2001. Dymola --- User’s Manual. Dynasim AB

Equa 2001. IDA Simulation Environment – User’s Guide. Equa Simulation AB (www.equa.se)

Equa 2002. IDA Indoor Climate and Energy – User’s Guide. Equa Simulation AB (www.equa.se/ice)

Hensen, J.L.M. 1995. Modeling coupled heat and air flow: ping-pong vs. onions. proc. 16th AIVC conference, Palm Springs, USA, 19-22 September, 1995.

IEEE. 1997. *Standard VHDL Analog and Mixed-Signal Extensions.* Technical Report IEEE 1076.1, IEEE, March 1997

ISO/DIS 7730. *Moderate Thermal Environments - Determination of the PMV and PPD indices and specification of the conditions for thermal comfort*, 1984

Jeandel A., F. Boudaud., and E. Larivière. 1997. ALLAN.Simuation release 3.1 description. M.DéGIMA.GSA1887. GAZ DE FRANCE

Kafetzopoulos, M.G. and K.O. Suen 1995. Coupling of thermal and airflow calculation programs offering simultaneous thermal and airflow analysis. Building Serv. Eng. Technol., 16, 1, pp 33-36, 1995.

Mundt, E. 1996. The Performance of Displacement Ventilation Systems. PhD Thesis, KTH, Stockholm, 1996

Oh M., and C.C. Pantelides. 1996. A modelling and simulation language for combined lumped anddistributed parameter systems. Computers and Chemical Engineering, **2**0, pp. 611--633, 1996

Piela P.C., T.G. Epperly, K.M. Westerberg, and A.W. Westerberg. 1991. ASCEND: An object-oriented computer environment for modeling and analysis: the modeling language. Computers and Chemical Engineering, 15**:**1, pp. 53--72, 1991.

Sahlin, P., E.F.Sowell, A Neutral Format for Building Simulation Models, Proc. of the IBPSA Building Simulation '89 conference, Vancouver, Canada, 1989

SEL 2000. TRNSYS, Version 15: A transient system simulation program. Solar Energy Laboratory

Stangerup, P. 1991. ESACAP: A Simulation Program and Description Language for Interdisciplinary Problems. proc, 1991 European Simulation Multiconference, Copenhagen, Denmark, June, 1991

Tiller, M., P.Bowles, H.Elmqvist, D.Brück, S. E.Mattson, A.Möller, H.Olsson and M.Otter 2000. Detailed Vehicle Powertrain Modeling in Modelica. Modelica Workshop 2000 Proceedings, pp.169-178

Walton G.N. 1997. CONTAM 96 Users Manual. NISTIR 6055, National Institute of Standards and Technology, USA.

Vuolle. M., A. Bring, P. Sahlin 1999 An NMF Based Model Library for Building Thermal Simulation, Proc. of the IBPSA Building Simulation '99 conference, Kyoto, Japan, 1999