Software for the Simulation of Power Plant Processes.

Part A: The Mathematical Model. + Part B.

Elmegaard, Brian; Houbak, Niels

Published in:
Proceedings of ECOS 2002

Publication date:
2002

Document Version
Early version, also known as pre-print

SOFTWARE FOR THE SIMULATION OF POWER PLANT PROCESSES PART B

Brian Elmegaard
Department of Mechanical Engineering
Technical University of Denmark (DTU)
Nils Koppels Alle, Building 402, DK-2800 Kgs. Lyngby, DENMARK
Phone: (+45) 4525 4169, Fax: (+45) 4593 5215, E-mail: be@mek.dtu.dk

Niels Houbak
Department of Mechanical Engineering
Technical University of Denmark (DTU)
Nils Koppels Alle, Building 402, DK-2800 Kgs. Lyngby, DENMARK
Phone: (+45) 4525 4154, Fax: (+45) 4593 5215,
E-mail: Niels.Houbak@mek.dtu.dk

ABSTRACT

Modelling of energy systems has been increasingly more important. In particular the dynamic behaviour is critical when operating the systems closer to the limits (either of the process, the materials, the emissions or the economics, etc.). This enforces strong requirements on both the models and their numerical solution with respect to both accuracy and efficiency. In this paper we give a survey on simulation of energy systems, from models and modelling, over numerical methods to implementational techniques. The paper is the second part of two papers covering important aspects of the different phases of modelling in general and modelling of an (energy) system. Part A, also gives a short introduction to robust numerical methods which it is strongly recommended to use. In this part, Part B, we present a survey of available, commercial and university simulators, a few important aspects of the implementation of the energy system simulator DNA and a short tricky example showing that too simple models may result in unexpected problems.

INTRODUCTION

During the last ten to twenty years, there has been a tremendous development of software for modelling and simulation of energy (or process) systems. The need for precise information about the behaviour of such systems has increased in connection with a higher degree of integration of processes, optimization of processes, getting closer to the limits of strength of the material due to temperature, pressure, or dimensions, or a need for better control of the processes due to for example quality of the product.

It is straightforward to divide these simulation programs into various groups: general simulators, application specific or component specific simulators.

When buying expensive equipment the customer may have included in the deal a simulator that could be used for training of the operators. This is the component specific case, where only the manufacturer may actually make changes in the underlying model. Often it is a non-trivial task to make serious changes to the overall structure of such a model because the model is tailored to the problem.

Application specific codes often offer a higher degree of freedom. In the design of the code, special attention has been paid to how problems traditionally are described within that specific application area. Programs for simulating the behaviour of electrical (or other types of) networks are of that nature.

General tools for simulating any system do not exist but there are tools (often equation based) where the user may type in almost any set of equations that he/she thinks describe the system. In most cases the solver may then actually produce reasonable results. From the above, it is clear, that the more the user wants the tool-producer to take responsibility for in the model, the less influence does the user have on the model to be solved.

This part B of the paper includes a survey of a number of the available tools and the inter-play between mathematical model and numerical methods. We give a description of the tabular representation of an energy system model as it is implemented in the component-based code DNA, and we exemplify whhat problems may develop when simple component models are connected in a system model.

AVAILABLE SIMULATION TOOLS

It is a fact that a very high number of codes for simulation of energy systems are available. This may be verified by a search on the interduct home page [2], which does not even contain all codes described in literature. On the other hand, it is also possible to verify that many of the tools:

- are specific to a narrow range of problems,
- are difficult to learn,
- do not handle model problems robustly,
- are not very well documented in literature, and/or
- are very expensive.

Before initiating a work on developing a simulation tool, Perstrup [22] made an assessment of codes available then. The main premise for the assessment was the desire for a tool with an extendible component library which might be applied for both steady state and dynamic simulation of all kinds of thermal energy systems, particularly power plants. One further specification following from this premise was that the code should be equipped with a well documented, efficient solver for AE, ODE, and DAE systems, i.e., a standard solver. The result of this assessment was that no code fulfilling the requirements existed. Though, this assessment since then has been maintained during the works described in [10, 20], it is not a complete overview of energy system simulation codes available. Below, we present the current status of our survey of available codes. Surveys with focus on overall featurs of different codes is given in [3, 23, 17] The presented codes are, to our knowledge, maintained and in use:

Aspen Plus is a commercial program which is developed for chemical processes and is often used in gas turbine applications. For steam plants its use is more limited. It is intended for steady state process simulation and uses a sequential solver [3, 17].

Camel is developed at the University of Rome [14, 15]. It is based on a well documented tabular description of the model and may be applied for both steady state and dynamic simulation. The latter is based on the assumption that dynamics may be calculated as a sequence of quasi-stationary steady-states, i.e., an explicit first-order Euler integration method. The solution method is sequential in the equations.

Code written by Consonni This is a steady-state solver which has been used in a variety of gas turbine and steam turbine system models [5, 6]

Cycle Tempo This program has a very high-

level graphical user interface, Guide,[25] and is suited for steady state simulations both for power and refrigeration systems. The solver of the program is a mixture between sequential and simultaneous methods, such that the linear balance equations are solved simultaneously, whereas the constitutive equations of the components are formulated explicitly in one unknown and evaluated in an inner loop for each outer loop evaluation of the balance equations [21]. Cycle Tempo is commercially available and is developed at Delft University of Technology.

DIMAP [1, 19] has been developed at University of Padova, Italy, and has been used in a number of studies of different cycles for steady state operation. The program has been extended with a graphical user interface. The program has a sparse matrix Newton solver which is applied in a special way to separated parts of the system of equations.

DNA is developed by several authors at the Technical University of Denmark. It is suited for both steady state and dynamic simulations [9, 10, 11]. Presently, there is no graphical interface, so DNA is a model description language. It has been integrated with the emacs editor and is free. The program uses a Newton solver to solve the system of equations, and it has a sparse linear equation solver. The robustness of the solver is improved by a preprocessing, sequential solution step. The differential equations are solved by a fourth order BDF method with variable step size and handling of discontinuities.

Dymola is a commercial solver for dynamic systems. It is the only simulation tool that is currently available for the simulation language Modelica, which is based on object oriented modelling and programming features. It has not yet been extensively used for energy systems, but the language is under development for this purpose and may be useful in the future. In Dymola, Modelica code is converted to C, and compiled to an executable which can be run interactively from Dymola.

Dymola includes several standard solvers for ODE/DAE's and handles discontinuities efficiently.

EES is a general equation solver with features for calculation of properties of a large number of fluids and solids. Experience shows that EES is very easy to use even for unexperienced users, e.g., students. EES is mainly intended for steady state simulation and it uses a robust implementation of the Newton method including sparse matrix techniques. It may be used for dynamic simulation as well, but the implemented ODE solver has limited efficiency.

ESMS is developed at the University of Florence[4]. It is suited for simulation of steady state processes and includes a graphical user interface. The solver is a specially implemented simultaneous linear equation solver.

GT Pro is a commercial program for steady state simulation of power plants. It uses a sequential solution method[17].

Gate Cycle is a commercial program for steady state simulation of power plants. It uses a sequential solution method[17].

Hysys is a commercial simulator for chemical processes for both steady state and dynamic simulation. It uses a sequential solution method.

ICAS is developed at the Technical University of Denmark and is a highly integrated software package for analysis of chemical process plants.

MMS/ACSL is a commercial simulator for dynamic systems [8]. MMS is a graphical extension to the ACSL modelling language – a preprocessor that generates

ACSL code. The ACSL model is translated to Fortran code which is compiled to an executable and may then be run. The solver is a sequential method, but does not handle algebraic loops [20].

Matlab/Simulink is a commercial system for simulation of dynamic systems in general. Thus, media properties are not available, but have to be implemented. Simulink is mainly intended for control system design and is not very well suited for closed loop systems, as is very often the case for energy system models. As a Matlab-based system it has access to all the mathematical analysis features available in Matlab. This includes several solvers for ODE systems.

Prosim is a commercial steady state simulation tool based on a sequential solver. It is so integrated into Autocad, that this is needed in order to run Prosim[3, 17].

Vissim is a commercial system for simulation of dynamic systems in general. Thus, media properties are not available, but have to be implemented. The program has several solvers for ODE's included.

Windali This program is developed at the Technical University of Denmark, as a code for dynamic simulation of DAE systems. It is closely integrated with the Windows platform and utilizes dynamic link libraries for integration between code and solvers. The program is available from [24]. It includes several standard DAE Solvers.

Some of the codes, e.g. [18], are the result of student or research projects and have as such often been developed for a special purpose and later extended to a more general applicability. This naturally leads to assumptions and neglects at the early development stages. This have to lead to limitations in extendibility. However, it is also our experience from the development of DNA, that even if the code has been developed with generality as a main aim, there will unavoidably be made decisions in the earlier stages of the development of the code, which will lead to difficulties when extending the code.

TABULAR MODEL REPRESENTATION IN DNA

DNA [9, 10, 20, 22] is an example of an energy system simulation tool which has both steady state and dynamic simulation features and has proven useful through several research and student projects involving simulation of e.g., steam power, gas turbines, fuel drying, pyrolysis and gasification, fuel cells, and heat exchanger networks [7, 10, 12, 13, 16].

DNA includes

- a modified Newton method solver for steady state models,
- an up to fourth order, variable step size BDF solver for dynamic simulation,
- an extendible component model library which is compiled into the code, and
- routines for calculation of state variable properties, transport properties and radiative properties of fluids and solids, e.g., ideal gas mixtures, water/steam, carbon dioxide and solid fuels and ashes.

DNA is a modelling language and the system model written by the user is compiled and simulated in one run.

During the compilation, the input is analyzed, information is re-organized, and stored properly in a set of tables. Thr simulation part is based upon standard solvers implemented with sparse matrix technique for efficiency.

The system models in DNA consist of components which have a number of branches connecting the component to its surroundings and parameters determining the characteristics of the component. The branches of two or more components are

connected at nodes in order to form a system. The tabular representation of the model makes it possible to automatically generate mass and energy balances for all components and nodes. Thus, the component modeller only has to implement the constitutive equations of the component type in a model. The tables also allows to exploit the sparsity of the system of equations and this feature is used in the solver.

From Physical to Mathematical Model

The mathematical model is a result of the physical model specified by the user as input to DNA. From this an internal, tabular representation of the model is generated. The table connects component and the connected nodes to the information about the component as is specified in the component library.

An important feature obtained by careful implementation of the tabular representation is that DNA will check for and issue errors if the system of equations resulting from the specified model does not have the same number of variables and equations, both in static parts and the dynamic parts of the system.

In order to have a complete tabular structure describing an energy system model, the input model specification generates a few more tables, e.g., for storage of gas mixture compositions.

A node-oriented description of the connections in the system is generated in order to check consistency.

From Mathematical to Numerical Model

The core tables for the solution process are generated from the above described tables. Three tables are necessary:

- The "'variable table"' holds information about all variables in the system. It has a column for each variable, describing the type of variable and the component it is connected to and its value.
- The "'residual table"' stores information about the equations (residuals) in the model. Each column holds information about an equation: The type of equation, the component or node it comes from, and the value.

  An equation is either conservation of energy or mass for a component or a node, a constitutive equation, or an equation relating dynamic variables to their time derivatives according to the Nordsieck formulation [20].
- The incidence matrix describing the entries of the Jacobian matrix. It is implemented using sparse matrix techniques as described in part A of the paper.

During the solution process these tables and the system table are frequently inspected when the residual values or the Jacobian of the system is to be calculated.

It should be noted that the above description does not cover all details of the implementation, and that it is elaborated further in [10, 20]. The source code is available from `http://www.et.dtu.dk/software/dna`.

In order to minimize the use of RAM in the computer, the complete set of tables in DNA is stored compactly in three one-column arrays, see [20].

LINEARLY DEPENDENT MODELS

The model displayed in figure 1 is a very simplified example of what may easily happen when applying standard, simple components to a simple (at first sight) problem. A heat source with negligible pressure loss provides hot water (100°C) for two parallely-coupled heat sinks both with a constant pressure loss of 9 bar. Both sinks cool the water to 10°C. The distribution of heat between the two sinks is fifty-fifty. A pump is used for raising the pressure to make the fluid circulate in the system.

Usually, a system model is built iteratively by inserting components one by one to create the complete model. In the present model it is easy to create the system as long

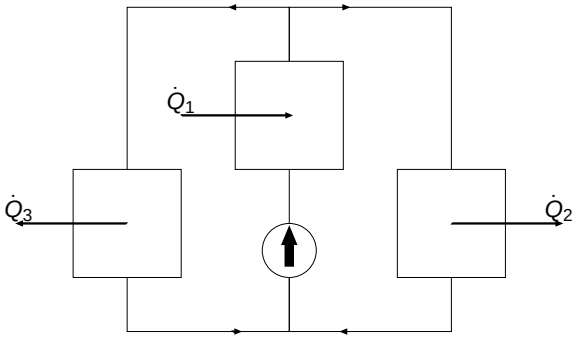

Figure 1: Sketch of a simple, problematic model

as the cycle is not closed, i.e., a system with inlet of water to the pump and outlet from the sinks may be made without problems. However, if the connection of the sink outlets to the pump inlet is made, the system can no longer be solved. The reason is that the constant pressure loss assumption of both the sinks generates two equations both specifying the absolute pressure into the pump. This is a linear dependency in the system, which cannot be solved. Another way to express this is: In the outer loop adding any number to the mass flows will make no difference.

Trying to simulate this system in DNA results in the error message, that one too many conditions on the system operation has been assigned. Removing any of the conditions does not solve the problem; DNA initiates the solution procedure and responds with an error message: The system of equations is wrong with some variables being overspecified, some being underspecified.

The solution is to decouple the pressures out of the two heat sinks. There are (at least) two ways to do this. Either, an insignificant mass flow dependency on the pressure loss may be introduced (a new component model is added to the simulator), or an expansion valve may be inserted between either of the sinks and the connection (an artificial component is introduced in the system model).

This example shows that even very simple problems may cause modelling problems, but also that modelling creativity may provide simple solutions to such problems. The problem is easily recognized here, but if it appears during the refinement or extension of a complete power plant model it is difficult to find.

CONCLUSION

In this paper, parts A and B, we have presented a survey of methods applicable in the implementation of energy system simulation software with focus on the implementation of internal model representation and numerical solvers. In this part B, we have presented a list of software and their main features with respect to the scope of the paper.

The DNA code was originally designed for large static problems; hence it was built with a Newton solver for the non-linear equations and sparse matrix technique for the linear equations. Em-phasis was put on having a complete component description including consistency checks, in one routine per component. Thus, evaluating the complete system model corresponds to calling the appropriate component routines in a systematic matter. The feature of including the composition of some of the fluids in the equation system was not in the original design but has been added later.

The extension of the code with dynamic capabilities did require several projects. In particular the implementation of the handling of discontinuities was a time-consuming task. It was in principle also this feature (or the lack of available discontinuity solvers) that caused an own-development of the DEA solver in DNA. Using the BDF method in a Nordsieck-formulation with a maximum (user-determined) order was simply a choice.

The conclusion is that all codes, including DNA, which have been presented in more detail, do have defiencies resulting from early assumptions in the implementation. These result in limitations in applicability and extendibility in further work on a code. We have described the implementation of DNA in detail in order to provide an insight in a way a code may represent a com-

plete model and solve it numerically.

Even if the code has been carefully implemented, a simple model of a simple energy system may result in problems and require the user to make innovative modelling efforts.

REFERENCES

[1] C. R. Altafini, A. Mirandola, and A. Stoppato. Analysis and simulation of an integrated gasification combined cycle power plant. In A. Bejan, M. Feidt, M. J. Moran, and G. Tsatsaronis, editors, *Efficiency, Cost, Optimization, Simulation and Environmental Aspects of Energy Systems and Processes*, pages 641–648, 1998.

[2] Anonymous. Home page of interduct. `http://www.interduct.tudelft.nl/`, 1994–2001.

[3] Mohsen Assadi, Per Rosén, and Niklas Ågren. Utvärdering av olika värmebalansprogram, (comparison of different heat balance programs). Technical Report LUTMDN/TMVK–3173–SE, Department of Heat and Power Engineering, Lund Institute of Technology, Sweden, 1995.

[4] C. Carcasci and B. Facchini. A numerical method for power plant simulations. *Journal of Energy Resources Technology*, 118:36–43, March 1996.

[5] S. Consonni and E. D. Larson. Biomass–gasifier/aeroderivative gas turbine cycles: Part b—performance calculations and economic assessment. *Transactions of the ASME—Journal of Gas Turbines and Power*, 118:516–525, July 1996.

[6] Stefano Consonni. Combined cycles for high performance, low cost, low environmental impact waste-to-energy system. Munich, Germany, May 2000. ASME.

[7] Pietro de Faveri Tron and Giacinto Carapelli. Analysis of an ifgt (indirectly fired gas turbine). Technical report, Department of Energy Engineering, Technical University of Denmark, 2000.

[8] S. Murthy Divakaruni. The aplication of simulation in large energy system analysis. *Modeling, Identification and Control*, 6(4):231–247, 1986.

[9] B. Elmegaard, N. Houbak, and B. Lorentzen. DNA—a network based energy system simulator. In Giampaolo Manfrida, editor, *FLOWERS'97—Florence World Energy Research Symposium*, pages 239–246. SGEditoriale, 1997.

[10] Brian Elmegaard. *Simulation of Boiler Dynamics – Development, Evaluation and Application of General Energy system Simulation Tool*. PhD thesis, Technical University of Denmark, 1999.

[11] Brian Elmegaard and Niels Houbak. Robust implementation of process simulators and their associated models. In G. Hirs, editor, *Proceedings of ECOS 2000*, pages 325–332, 2000.

[12] Brian Elmegaard and Bjørn Qvale. Analysis of indirectly fired gas turbine for wet biomass fuels based on commercial micro gas turbine data. Submitted to ASME IGTI Turbo Expo 2002, Amsterdam, June 2002.

[13] Brian Elmegaard, Bjørn Qvale, Giacinto Carapelli, and Pietro de Faveri Tron. Open-cycle indirectly fired gas turbine for wet biomass fuels. In *Proceedings of ECOS '01*, pages 361–368, 2001.

[14] Marco Francesco Falcetta and Enrico Sciubba. A computational, modular approach to the simulation of powerplants. *Heat Recovery Systems and Combined Heat and Power Production*, 15:131–145, 1995.

[15] Marco Francesco Falcetta and Enrico Sciubba. Unsteady numerical simulation of combined cycle plants. In R. Cai, M. J. Moran, S. Zhangz, and Y. Xiao, editors, *Thermodynamic Analysis and Improvement of Energy Systems*, pages 224–231. Chinese Society of Engineering Thermophysics and American Society of Mechanical Engineers, Beijing World Publishing, 1997.

[16] Nicola Gelli, Giovanni Sarti, and Marco Donati. Biomass fuelled power plants. Technical report, Technical University of Denmark, Department of Energy Engineering, 2000.

[17] I. Giglmayr, M. Nixdorf, and M. Pogoreutz. Vergleich von software zur thermodynamischen proceßrechnung. Forschungsvorhaben 177, VGB, Technische Universität Graz, Technische Universität München, March 2000.

[18] A. Korving. Cycle–tempo. `http://www-pe.wbmt.tudelft.nl/ev/cycle/cycle.html`, 1999. Section Thermal Power Engineering, Delft University of Technology.

[19] A. Lazzaretto et al. Dimap—a modular computer code for thermodynamic exergetic and thermoeconomic simulation of energy systems. In R. J. Krane, editor, *Thermodynamics and the design, analysis and improvement of energy systems. The 1995 ASME, Proc. of the International Mechanical Engineering Congress and Exposition*, volume AES-35, pages 119–126. ASME, American Society of Mechanical Engineers, 1995.

[20] B. Lorentzen. *Power Plant Simulation*. PhD thesis, Technical University of Denmark, Laboratory for Energetics, 1995.

[21] J. A. Miedema. *Cycle; a General Computer Code for Thermodynamic Cycle Computations*. PhD thesis, Delft Unversity of Technology, 1981.

[22] Claus Perstrup. Analysis of power plant by application of network theory (in danish). Master's thesis, Technical University of Denmark, Laboratory for Energetics, 1991.

[23] Robert M. Privette. *Developments in the Design of Thermal Systems*, chapter Computer-aided process design trends in the power industry, pages 16–39. Cambridge University Press, 1997.

[24] Morten Skovrup. Windali home page. Internet: `http://www.et.dtu.dk/software/windali`.

[25] TNO Milieu, Energie en Procesinnovatie and Delft University of Technology. *CYCLE–TEMPO, Thermodynamic Energy systems Massflow calculation of POwer processes*, release 3.21 edition.