Developments to the TRNSYS Simulation Program

A. Fiksel

J. W. Thornton

S. A. Klein

W. A. Beckman

Solar Energy Laboratory,
University of Wisconsin,
Madison, WI 53706

The TRNSYS simulation program is a sequential-modular transient simulation program which is widely used for both solar and nonsolar simulation studies despite its near 20-year-old origins. There have been many revisions to TRNSYS between its original release and the current version 13.1 but they have maintained the same sequential computational scheme for solving simultaneous algebraic and differential equations. TRNSYS 14 has recently been developed which implements a more robust method for solving simultaneous sets of nonlinear equations. The new computational scheme utilizes equation blocking to improve convergence properties and it handles discrete control decisions in a manner which promotes convergence in the iterative calculations. The new computational scheme allows TRNSYS 14 to solve problems which were either difficult or impossible to solve in previous versions as well as backwards problems, for which the output of a component is specified and the input must be determined. The computation scheme incorporated into TRNSYS 14 is described in this paper and its capabilities are illustrated with several examples.

Introduction

Solar energy systems exhibit a nonlinear dependence upon the weather on both short (e.g., hourly) and long (seasonal) time scales. As a result, simulations provide the only practical means of optimizing the design of solar energy systems. The TRNSYS program is a sequential-modular transient simulation program developed specifically for solar energy system analyses (Klein et al., 1975, 1976). With TRNSYS, each system component is modeled with a separate FORTRAN subroutine which is either taken from the TRNSYS library or supplied by the user. The component models can be analytical or empirical representations of the component performance. The TRNSYS program allows the user to connect the components in a specified manner and then solves the resulting set of simultaneous algebraic and differential equations. The modular format and the ability for the user to provide his or her own component models provide TRNSYS with general simulation capability applicable to almost all transient system problems. This generality, along with the continuous program technical support, are the major reasons that TRNSYS has been and continues to be widely used for both solar and nonsolar simulation studies.

There have been many revisions to TRNSYS since its original release in 1975 but all of these revisions maintained the same sequential computational scheme for solving simultaneous algebraic and differential equations. Version 14, however, implements quite different solution algorithms which are described in this paper. To appreciate why these changes were needed, however, it is first necessary to review the computation methods used in TRNSYS 13 and earlier versions.

Contributed by the Solar Energy Division of The American Society of Mechanical Engineers for publication in the ASME Journal of Solar Energy Engineering. Manuscript received by the ASME Solar Energy Division, Sept. 1993; final revision, June 1994. Associate Technical Editor: J. H. Morehouse.

Computational Scheme in TRNSYS 13

In TRNSYS, a simulation model is created by developing and interconnecting models of the system components. The component models, which may be empirical or physical, describe the component performance with algebraic and/or differential equations. The performance of a system is the combined performance of its components. Three distinct types of information are associated with each component model: parameters, inputs, and outputs. Parameters provide information which does not change with time, such as the physical size of the component. Inputs provide information which may change with time, such as temperatures, flow rates, energy flows, etc. The component models, written as a FORTRAN subroutines, use the parameters and inputs to calculate outputs as functions of time. The component models are connected together to form an information flow diagram for the system by linking the outputs of one component to the inputs of another. As example information flow diagram is shown in Fig. 1 for a simple solar space heating system. Each box represents a component model. The input-output connections which transmit information between component models, represented as lines in Fig. 1, are analogous to the pipes and wires which interconnect the real components.

The information flow diagram of a system may exhibit recyclic information flow in which the outputs of a component model lead to other system components and then back to the original component. In this case, an interative technique is needed to solve the resulting set of equations. When recyclic information flow exists, TRNSYS 13 uses a successive substitution iteration scheme until all of the input variables in the recyclic loop converge to a tolerance specified by the user.

Differential equations may either be solved analytically by conversion into an algebraic form or numerically using the integrator built into TRNSYS 13. The differential equations often require a simultaneous solution with the algebraic equa-

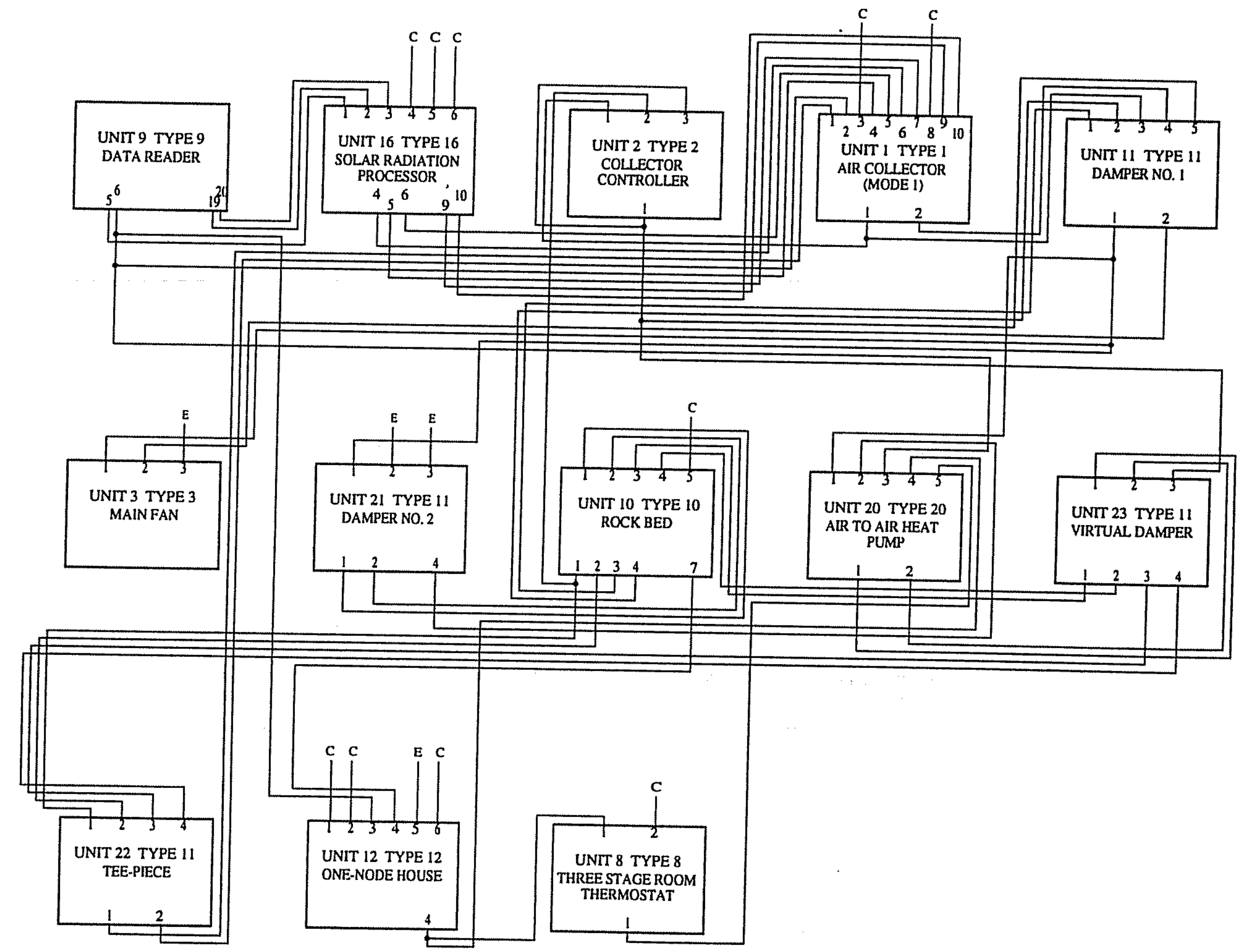

Fig. 1 TRNSYS information flow diagram for a solar space heating system

tions. TRNSYS 13 provides several integration algorithms. The original scheme is a first-order predictor-corrector method in which Euler's method is used for prediction and the trapezoid rule is used for the correcting step. The advantage of a predictor-corrector integration algorithm for solving simultaneous algebraic and differential equations is that the iterative calculations needed to solve the differential equations occur at a fixed value of time. As a result, the solutions to the algebraic equations can converge by successive substitution as the iteration required to solve the differental equations progresses. TRNSYS 13 also provides an optional fourth-order predictor-corrector algorithm.

An important part of the TRNSYS 13 solution scheme is that only those component models whose inputs (which are outputs from other components) or differential equations have not converged to within specified tolerances are called during the iterative calculations. In this manner, the computational effort needed to solve the algebraic and differential equations during each time step is minimized.

Problems with the Computational Scheme in TRNSYS 13

Although relatively simple, the computational scheme used in TRNSYS 13 has proven to be reliable and efficient for simulating systems with energy storage such as solar domestic hot-water systems. These systems typically have less than 50 coupled differential equations and 100 simultaneous nearly linear algebraic equations with few controller decisions. The limitations of the computational scheme become apparent when TRNSYS 13 is used to solve sets of nonlinear algebraic equations without differential equations. Equations of this type occur in systems for which the energy storage is negligible, such as for a photovoltaic array directly coupled to a load or a refrigeration system operating at steady-state conditions. The successive subtitution solution method used in TRNSYS 13 does not efficiently solve nonlinear algebraic equations and it may, in fact, not be able to find a solution if the equations are highly nonlinear. TRNSYS 13 has no formal means of supplying suitable guess values of the INPUTS at each timestep (except for using the solution at the previous timestep) and it has no provision to bound the values it is trying to determine.

An Accelerate command was added to version 13 to improve vergence in problems with recylic information flow. The Accelerate command allows the user to break a selected input-output connection and replace it with a single-variable Newton's method solution algorithm. Although useful in some circumstances, the Accelerate command has been unsatisfactory for two reasons; (1) it requires the user to identify the appropriate input-output connection and (2) it implements a single-variable solution method when in many situations, a multiple-variable method is needed. Another way in which numerical convergence problems have been handled in the past is by the user coding convergence-enhancing techniques within the component models. For example, "combined-component" models have been developed for TRNSYS wherein the nonlinear equations describing two or more pieces of equipment are solved internally in a single component model. Although "combined-component" models may eliminate numerical problems, they reduce component modularity and require the user to implement solution techniques, thereby defeating the original purpose of TRNSYS.

The distinction between inputs and outputs made in TRNSYS 13 places a limitation on the generality of the component models. When a model is formulated, a decision must be made as to what are the inputs and what are the outputs. For example, a model may be developed to calculate an energy flow for a given mass flow rate and temperatures. However, in a different application, the energy flow rate may be known and the model must calculate the mass flow rate. The same physical model is used in either situation, but the inputs and outputs are different. TRNSYS 13 requires different component models (or modes) to handle the change in information flow. It would be most useful if TRNSYS could backsolve to determine the inputs for given outputs.

TRNSYS 14 Computational Scheme

The limitation of the TRNSYS 13 computational scheme have provided an incentive to significantly revise the executive routines of the program. A large library of component models have been developed for TRNSYS. A major consideration in this revision has been to maintain compatability with the existing component library and therefore with the input-output-parameter information flow classification.

The equations within any TRNSYS component model can be reduced to the following general form:

$$\dot{X}_i = D_i(X_i,\ U_i,\ t) \tag{1}$$

$$X_i = F_i(U_i,\ t) \tag{2}$$

where U_i and X_i are, respectively, the vector of inputs and outputs for the module, t is time, and D_i and F_i are functions representing the differential and algebraic equations, respectively. An analogous set of equations can be written for the combined set of equations in all components,

$$\dot{X} = D(X,U,t) \tag{3}$$

$$X = F(U,\ t) \tag{4}$$

where U and X are vectors containing the inputs and outputs for all components. Every input in a TRNSYS model is either an output from another component or a specified value. This additional information can be written as a matrix equation

$$\mathbf{A}U + \mathbf{B}X + C = 0 \tag{5}$$

where **A** and **B** are coupling matrixes with element values of 0 and $+/-1$, and C is a vector of boundary conditions holding the values of inputs which are set to fixed constants or known functions of time. The system of algebraic differential Eqs. (3)–(5) can now be solved to determine $U(t)$ and $X(t)$. This computational scheme is similar to that used in the IDA modular system simulation program (Sahlin, 1991).

The number of simultaneous nonlinear equations resulting from this computational scheme can become very large. To minimize computational effort, it is necessary to employ methods for decreasing the number of simultaneous equations. Some TRNSYS inputs are always known, either because they are constants or they depend explicitly only on time. Moreover, it is possible for all inputs in some TRNSYS components to be of this type. In this case, the outputs of these components can be calculated independently of other components. Once the outputs of these components are known, the inputs to which they are connected are known, possibly allowing additional components to be evaluated independently. This process is repeated until only the components which require simultaneous solution remain. This process is a first step in equation blocking in which a large set of equations is broken into smaller sets which are more easily solved.

The remaining sets of equations are solved numerically. There are a number of well-developed numerical methods for these types of equations (Gerald and Wheatley, 1984, Broyden, 1965). In TRNSYS 14, differential equations are solved by a backwards differential method (Gear, 1967) so that the combined algebraic and differential systems can be converted to an algebraic system which can be solved at each time step. This method is extremely robust, even for stiff problems. In addition, this method allows use of variable time steps, although TRNSYS 14 does not currently employ variable time steps. Before solving the resulting system of algebraic equations, TRNSYS 14 numerically calculates the Jacobian matrix and permutes it to lower triangular form. This permutation facilitates equation blocking which breaks the original system into smaller sets of equations which can be solved more efficiently (Duff et al., 1986). Each block of equations is solved using Powell's method (Powell, 1970; Duffie and Beckman 1991). This method combines Newton's method and the steepest descent approach and is one of the more robust and efficient algorithms for solving nonlinear equations.

A major advantage of the new computational scheme is that it does not matter whether inputs or outputs are specified as long as a valid set simultaneous equations are defined. As a result, TRNSYS 14 has the ability to solve backward problems, i.e., problems in which one or more outputs are specified and the corresponding inputs must be determined. An Outputs command, similar in function to the Inputs and Parameters commands, has been added to TRNSYS 14 to facilitate specified output problems. An example of a backward problem is provided in the Examples section.

Unfortunately, there is also a disadvantage of the new computational scheme. Because the new solver determines the Jacobian matrix numerically. TRNSYS 14 usually requires more component calls each time step than the scheme used in TRNSYS 13—provided that TRNSYS 13 is able to solve the problem. The computational scheme in TRNSYS 14 is far more robust and is consequently able to solve a greater variety of problems without experiencing computational difficulties. Because some problems can be more efficiently solved with the older computational scheme, both computational schemes are made available to the user in TRNSYS 14. A SOLVER command has been added to TRNSYS to select the computational scheme.

Handling of Discontinuities and Control Decisions

A problem which has plagued TRNSYS 13 and other general equations solvers in the past is the presence of discontinuities or discrete states, particularly when they demonstrate hysteresis properties. Many existing TRNSYS components have equations of this type. The most common example is the on/off Differential Controller provided as Type 2 in the standard TRNSYS library. This component calculates a control signal, which is either 0 or 1, based on the differences between two input temperatures and the control signal from the previous calculation. The functional relation between the controller output and its inputs is shown in Fig. 2. If the temperature difference is less than ΔT_{off}, the control signal is 0. If the temperature difference is greater than ΔT_{on}, the control signal is 1. However, if the temperature difference is between ΔT_{off} and ΔT_{on}, the output control signal is either 0 or 1, depending on its previous value. This type of mathematical behavior can lead to oscillations in the calculations preventing a numerical solution to the system of equations. In fact, there may not be a stable numerical solution, depending on the selected values of ΔT_{off} and ΔT_{on}.

There are many examples of on/off controls is the TRNSYS component library, such as the Flow Diverter (Type 11), the Energy/Degree-Day Space Heating or Cooling Load (Type 12) and the Pressure Relief Value (Type 13). Other examples of discontinuities which appear in TRNSYS component models

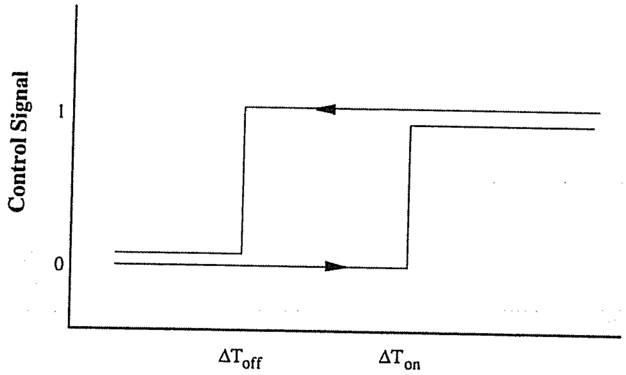

Fig. 2 TRNSYS Type 2 controller with hysteresis properties

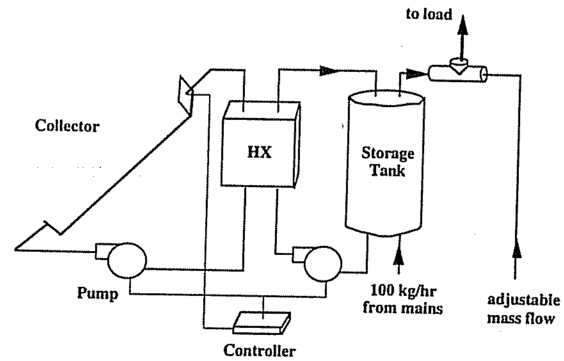

Fig. 3 Simple water heating system with variable water flow rate

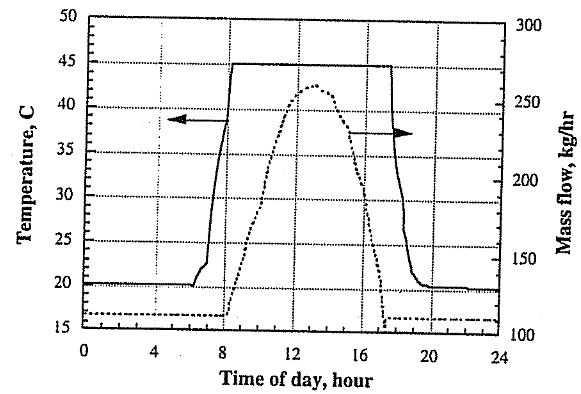

Fig. 4 Calculated water flow rates and temperature for the system in Fig. 3

are the shift from turbulent to laminar flow regime which causes a discrete change in a heat-transfer coefficient or friction factor, and the internode flow control logic for a stratified water storage tank. During the iteration process the component outputs can switch states resulting, in effect, in a different set of equations with a different Jacobian. TRNSYS 13 handles the convergence problems which result from this behavior by "sticking" the state after a specified number of iterations. For example, the Type 2 on/off differential controller has a parameters called NSTK which is the number of state changes allowed before the controller state is frozen at its current value. Although freezing the controller state eliminates most convergence problems, it may lead to incorrect results in some cases, particularly in short-term simulations.

TRNSYS 14 provides a very different approach for handling these problematic discontinuities. The TRNSYS executive program, rather than the component models, directly controls discrete variables. At the start of each time step, the values of all discrete variables are known. TRNSYS 14 forces these values to remain at their current state until a converged solution to the equations is obtained or an iteration limit is encountered. During these calculations, the component models calculate the "desired" value of the discrete variables but they do not actually change the settings. After a converged solution is obtained or the maximum number of iterations is exceeded, TRNSYS 14 compares the current discrete variable values with the 'desired' values. If they do not differ and a converged solution has been obtained, the calculations are completed for the time step. Otherwise, TRNSYS 14 changes the values of the discrete variables to the "desired" setting and repeats this process, taking care to not repeat the calculations with the same set of discrete variables used previously. If a solution is not obtained after all combinations of discrete variables have been tried, TRNSYS 14 issues a warning message and continues on to the next time step. This method of dealing with discrete variables eliminates numerical oscillations between iterations and finds the correct solution to the equations, provided a solution truly exists. Changes in the component models utilizing discrete variables are required to take advantage of this computational scheme so that the TRNSYS solver, rather than the component model, makes the control decision. These changes have been made in many of the TRNSYS library components.

Examples

Solar Water Heating System. The major advantage of the new computational scheme in TRNSYS 14 is that it allows TRNSYS to solve problems that could not be solved previously without the user providing some form of convergence control. A second advantage, equally important, is that the new computational scheme allows solution of backwards problems, eliminating the need to provide separate component modes or even different component models when the information flow in a problem is changed. The following two simple examples illustrate these advantages.

The first example shown in Fig. 3 is the typical solar domestic hot water system with a collector-tank heat exchanger. When solar radiation is available, a fluid is circulated through the solar collector at a constant mass flow rate. The collected energy is transferred through a heat exchanger to a storage tank. 100 kg/hr of water constantly flows into the bottom of the storage tank. Hot water removed from the upper part of the tank is mixed with cold supply water in the tempering valve at a mass flow rate sufficient to maintain the temperature of the water going to the load below a specified maximum.

All of the components required to simulate this system exist in the standard TRNSYS library. However, the tee-piece component model has been developed to calculate the outlet temperature and mass flow rate given known mass flow rates and temperature for the inlet streams. In this case, it is necessary to calculate the mass flow rate of the mains supply stream in order to control the water outlet temperature. This is an example of a "backwards" problem in which an output is known and it is necessary to calculate an input. This problem could not be solved with the existing tee-piece model in the TRNSYS library using previous versions of TRNSYS. Results of the simulation with TRNSYS 14 for a day in August in Sacramento, CA are shown in Fig. 4 for controlled water outlet temperatures of 45°C.

The ability provided in TRNSYS 14 to solve backwards problems is very convenient since it does not require any modification to the existing components. In many cases, however, backward problems of this type do not have a solution or they may have a solution for only certain hours of the day. In the system shown in Fig. 3, for example, it is possible to keep the

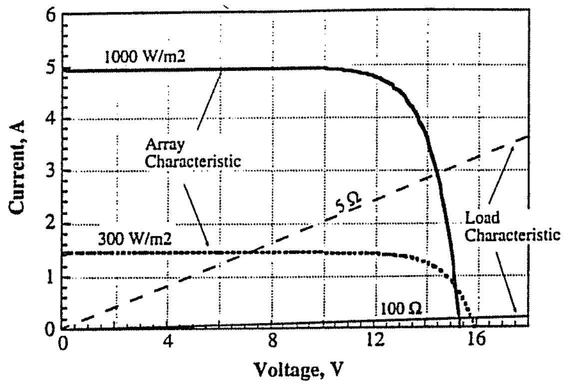

Fig. 5 Current-voltage curves for PV array at two radiation levels and two resistance loads

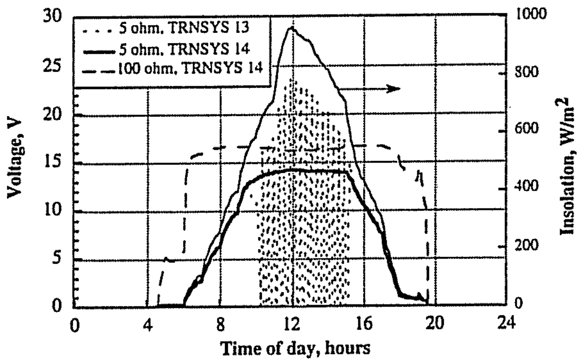

Fig. 6 The dependence of PV array voltage on time for different loads

outlet water temperature above 45°C only between 8:00 a.m. and 6:00 p.m. as seen in the simulation results in Fig. 4. A solution to the backwards problem does not exist during the remainder of the day. If TRNSYS 14 can not find a solution to the backward problem, it will issue a warning message and proceed to solve the forward problem in which the inputs are known. In this case the values of the unknown inputs are set to their initial values and the outputs are calculated corresponding to these inputs values. The result of this computational scheme is evident in Fig. 4 in which the mains water flow rate through the tee-piece was set to 10 kg/hr outside of the interval between 8:00 a.m. and 6:00 p.m. resulting in a total mass flow rate of 110 kg/hr.

Photovoltaic System. A photovoltaic array directed coupled to a resistance load results in a rather difficult problem for TRNSYS because the system has no storage capacity and therefore no differential equations resulting in a set of highly coupled nonlinear algebraic equations. The photovoltaic array is modeled using Eqs. (23.2.6)–(23.2.8) from (Duffie and Beckman, (1991). Current-voltage plots for the PV array at two radiation levels are shown in Fig. 5. Also shown in Fig. 5 are the straight-line current-voltage relations for two resistance loads.

At a specified time, and therefore at known solar radiation and ambient temperature, the photovoltaic array current-voltage characteristics are known. It is now necessary to find the current and voltage at the intersection point of the load and array curves. For most insolation levels and loads, this algebraic system is very nonlinear, as evident in Fig. 5. The large slope of the array current-voltage curve causes difficulty in finding a numerical solution. Photovoltaic systems have been simulated previously in TRNSYS 13 (Eckstein, 1990). However, in earlier studies, it has been necessary to add convergence enhancing code to the component models, or to combine the array and load models into a single component. In this example, however, the photovoltaic array and the load are each modeled as separate components using only the governing current-voltage equations. The results of a simulation for a 24-hour period with 1/10th hour time steps on June 1 in Madison, WI are shown in Fig. 6 using both TRNSYS 13 and TRNSYS14.

As the insolation increases, TRNSYS 13 is unable to solve this problem even for small loads. The calculated performance is unstable, oscillating above and below the true solution every other time step. TRNSYS 14, however, experiences no difficulties in solving this problem for a wide range of insolation, ambient temperature, and loads.

Conclusions

TRNSYS, originally developed in the early 1970s, has been widely used in simulation studies of both solar and nonsolar systems. The computational scheme in TRNSYS was developed for a particular class of algebraic and differential equations. This computational scheme is either inefficient (requiring many iterations and excessive computer time) or fails to solve problems involving highly-coupled algebraic equations. The computational scheme also fails to properly handle some discrete variable problems, such as those involving a controller with hysteresis. For these reasons, a new computational scheme has been added to TRNSYS 14. The new computational scheme allows TRNSYS to successfully simulate systems which could not be simulated with earlier versions. In addition, the new computational scheme handles backwards solution problems in which the outputs are known and inputs must be determined. The backwards solution problem capability eliminates the need for separate components or modes and should thereby simplify component model development.

References

Broyden, C. G., 1965, "A Class of Methods for Solving Nonlinear Simultaneous Equations," *Math. Comp.*, Vol. 19, pp. 577–593.

Duff, I. S., Erisman, A. M., and Reid, J. K., 1986, *Direct Methods for Sparse Matrices*, Oxford Science Publications, Claredon Press.

Duffie, J. A., and Beckman, W. A., 1991, *Solar Engineering of Thermal Processes*, 2nd ed., John Wiley and Sons, New York.

Eckstein, J. H., 1990, "Detailed Modelling of Photovoltaic System Components," M. S. Thesis, Department of Mechanical Engineering, University of Wisconsin, Madison, WI.

Gear, C. W., 1967, "The Numerical Integration of Ordinary Differential Equations," Math. Comp., Vol. 21, pp. 22–49.

Gerald, C. F., and Wheatley, P. O., 1984, *Applied Numerical Analysis*, Addison-Wesley, p. 190.

Klein, S. A., et al., 1975, "TRNSYS-A Transient System Simulation Program "User's Manual," Solar Energy Laboratory, University of Wisconsin, Madison, WI.

Klein, S. A., Beckman, W. A., and Duffie, J. A., 1976, "TRNSYS—A Transient Simulation Program," *ASHRAE Tranactions*, Vol. 82, Part 1.

Klein, S. A., et al., 1975, T "TRNSYS—A Transient System Simulation Program "User's Manual," Solar Energy Laboratory, University of Wisconsin, Madison, WI.

Powell, M. J. D., 1970, "A Hybrid Method for Nonlinear Equations," *Computer Journal*, Vol. 12, pp. 87–114.

Powell, M. J. D., 1970, "A Fortran Subroutine for Solving Systems of Nonlinear Algebraic Equations," *Computer Journal*, Vol. 12, pp. 115–161.

Sahlin, P., 1991, "IDA—A Modeling and Simulation Environment for Buildings, Application," ITM Report NO. 1991:2, Goteborg, Sweden.