TACS Enhancements for the Electromagnetic Transient Program

R.H.Lasseter, Fellow, IEEE and J. Zhou
University of Wisconsin-Madison

Abstract- Transient Analysis of Control Systems (TACS) of the Electromagnetic Transient Program (EMTP) has been enhanced in many ways. There are major changes in methods of ordering components to minimize the introduction of time step and history term errors. Initialization algorithms have been greatly enhanced to allow multiple frequency initialization. An "IF THEN ELSE" control structure and the use of user written FORTRAN routines are among the important new features available to users of TACS.
Keywords: TACS; EMTP; Control Systems; Simulation.

I. Introduction

The Electromagnetic Transient Program (EMTP) was originally developed in the late 1960's by Hermann Dommel [1]. The program was further developed at Bonneville Power Administration and has become an important industrial tool for analyzing power system transients. Currently the EMTP Development Coordination Group made up of utilities and companies from North America, Europe and Japan coordinate effort on EMTP development. The work discussed in this paper is part of this effort.

The Transient Analysis of Control Systems (TACS) was introduced to the EMTP program [2] by L. Dube in 1977. This addition allows a general representation of control systems in EMTP models. This is achieved through various system elements, such as transfer functions, algebraic functions and special devices that can be connected in an arbitrary manner to model a given control system. TACS has become important for the study of high voltage direct current transmission systems, static Var compensators, and generators when the transient responses of their respective control systems are important to the problems at hand.

This paper discusses the philosophy and implementation of enhancements to TACS. These enhancements are to be incorporated in DCG/EPRI version 3.0 of the EMTP. A different approach to basic TACS problems has been recently described by Dube and Bonfanti (3).

This paper was presented at the 1993 IEEE Power Industry Computer Application Conference held in Scottsdale, Arizona, May 4 – 7, 1993.

II. Basic TACS Solution Method

The solution methods used in previous releases of TACS assumes that the principle building block is a nth order transfer function that can be expressed as a system of first-order differential equations. The use of the trapezoidal rule of integration results in a simple algebraic equation where the derivatives of the input and output variables are eliminated. This produces a relationship for the nth order transfer function of the following general form where $u(t)$ and $x(t)$ are respectively the input and output of a given transfer function.

$$cx(t) = Kdu(t) + hist(t - \Delta t) \quad (1)$$

The coefficient K is the gain of the transfer function while c and d are calculated from the coefficients of the transfer function using recursive relationships [2]. The history term is calculated from the previous time-step solution.

A complete control system, with many elements, results in a set of equations that may be expressed in a matrix form.

$$[A_{xx}]\,[x] + [A_{xu}]\,[u] = [hist] \quad (2)$$

In this equation $[x]$ is a vector of the unknown variables or states and $[u]$ is a vector of known sources. To solve this system of equations, TACS performs a triangular factorization on $[A_{xx}]$, *and* $[A_{xu}]$. For each time step solution the $[hist]$ and source terms are formed by forward substitution followed by back substitution to obtain the system states $[x]$.

This method results in simultaneous solution for systems composed solely of transfer function blocks. For the case of a single limiter on a transfer function a simultaneous solution can also be found. This is achieved by ordering the system such that the output of the element with the limiter is the first variable found in the back substitution.

Simulation of control systems components other than transfer functions are included in TACS. These components include FORTRAN-like functions and expressions, logical functions, and special devices. In general these elements must be viewed as non-linear function blocks that are not directly included in the simultaneous solution discussed above. If these non-linear elements are included in a feedback loop, TACS breaks the loop by inserting a time delay. In most cases these time delays cause few problems. The existing TACS codes, through DCG/EPRI Version 2.0, also introduces additional time delays and in some situations incorrectly calculates the history terms. These issues can best be explained with a simple example.

Consider the example shown in Figure 1. This example includes a single transfer function F3 and simple gains F1 and F2. The response of this system to a step input is shown in Figure 2 under the label "simultaneous solution." The triangular form of the matrix equation for this system is shown as equation (3).

$$\begin{aligned} A_{11}X_1 + A_{12}X_2 + 0 + 0 &= hist_{F1} \\ 0 + A_{22}X_2 + A_{23}X_3 + 0 &= hist_{F2} \\ 0 + 0 + \bar{A}_{33}X_3 + A_{3s}U_s &= \overline{hist}_{F3} \end{aligned} \quad (3)$$

The first row expresses the relationship between the input X_2 and the output X_1 for element F1. The same follows for elements F2 and F3. Due to triangulation the coefficient $\bar{A}_{33}$ is different from the original coefficient for element F3. More important is the history term, $\overline{hist}_{F3}$ which is dependent on all the states of the system, X_1,X_2, and X_3. Clearly when the history terms and the source U_S are known the output of F3 can be found. This is followed by calculating the outputs of F2 and F1. Once all states are known the history terms for the next time step are calculated. This is a very powerful method for finding simultaneous solutions for an arbitrary circuit of transfer functions.

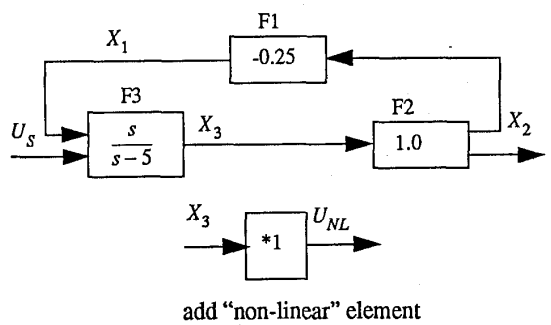

FIGURE 1. Simple Feedback Control System

More practical systems require the use of some non-linear elements. Non-linear functions are not directly included in the matrix form shown in equation 2. They are calculated as their inputs become available. Simultaneous solutions can be found for systems with non-linear elements provided these elements are not part of any feed back loop. For cases where there is feedback, which includes non-linear elements, simultaneous solutions can not be found without iteration at each time step. This can be costly in computation time and is not used by TACS. The solution method used by TACS is to decouple the feedback paths by introducing a time delay into the loop.

To understand the problems introduced by non-linear elements the circuit in Figure 1 is used. A dummy "non-linear" element is introduced which does not change the response of the circuit. This element is introduced between element F3 and F2. The current TACS treats this element as a general non-linear element. In this case the output of the "non-linear" element is assumed to be a known source, U_{NL}, not a state to be found. The resultant set of equations is shown in (4).

$$\begin{aligned} A_{11}X_1 + A_{12}X_2 + 0 + 0 &= hist_{F1} \\ 0 + A_{22}X_2 + 0 + A_{23}U_{NL} &= hist_{F2} \\ 0 + 0 + A_{33}X_3 + A_{3s}U_s + \bar{A}_{23}U_{NL} &= \overline{hist}_{F3} \end{aligned} \quad (4)$$

The two sets of equations (3) and (4) are for the same system. The equation for F1 is not changed while F2 and F3 have different forms due to treating the output of the "non-linear" block as a source. When the equations are solved using back substitution a time delay is introduced. To find X_3 the value used for "source" U_{NL} is from the previous time step calculation. This has the effect of decoupling the loop at the input to F2 and solving the system in a sequential manner.

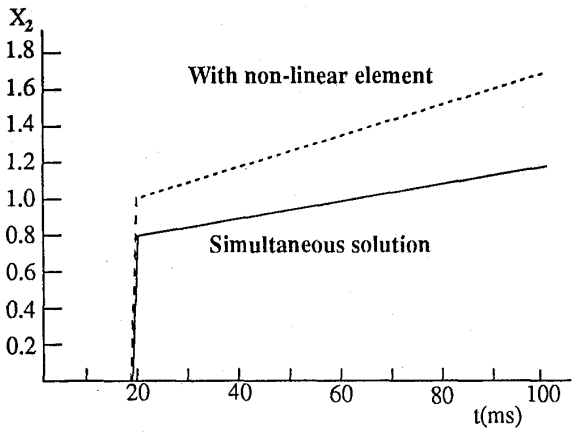

FIGURE 2. Response of System to a Step Input.

The results of including this "non-linear" element are also shown in Figure 2. This example shows a much more severe problem than a single time step delay. If the calculation was correct the added time delay would result in a solution that would oscillate about the simultaneous solution, Figure 3. In this example the feedback from F1 is lost resulting in a very different solution. This problem is due to how the history terms are calculated. In particular the history terms are calculated after all the states are found. This can become a problem when non-linear elements are introduced.

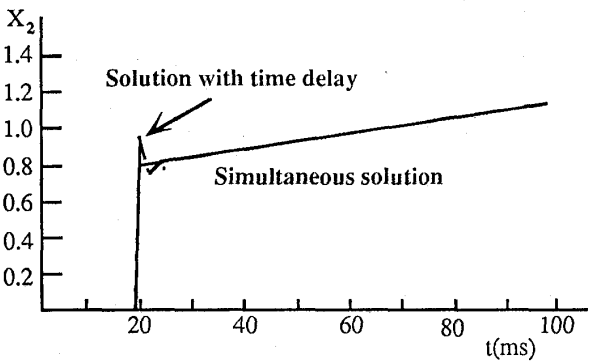

FIGURE 3. Correct Response with time delay

In this example, equation (4) indicates that X_3 is found first using the value for U_{NL} from the previous time interval. Once X_3 is known the output of the non-linear element is calculated. This new value of U_{NL} is used to find the other states followed by calculation of the new history terms. This provides an inconsistent history term for F3. Its state was found using a previous value for U_{NL}, but its history term was calculation using the updated value for U_{NL}. Problems with unnecessary time delays and history term errors have resulted in a new approach to finding TACS solutions.

III. New Solution Method for TACS

As discussed in Section II, the current TACS has problems with excessive time step delays and the calculation of history terms when non-linear elements are introduced. The objective of the current ordering method was to reduce fill-ins, saving computer storage and reduce computational time. This is still the objective in the EMTP program. The objective of the new ordering algorithm for TACS is to provide a sequential ordering of all elements that minimizes the time delays and correct errors in the history term calculation. This ordering method should also be independent of the type of element and depend only on the connections between the elements.

Sequential ordering has the problem of introducing time delays in a feedback loop when it is possible of find a simultaneous solution using the matrix methods discussed in the previous section. In this case the new algorithm must also search for sub-blocks within the sequential ordering that can be solved in a simultaneous manner. For ordering purposes these sub-blocks can then be handled as a single block, or "super block." To better explain the new algorithm the example test system shown in Figure 4 will be used as an example.

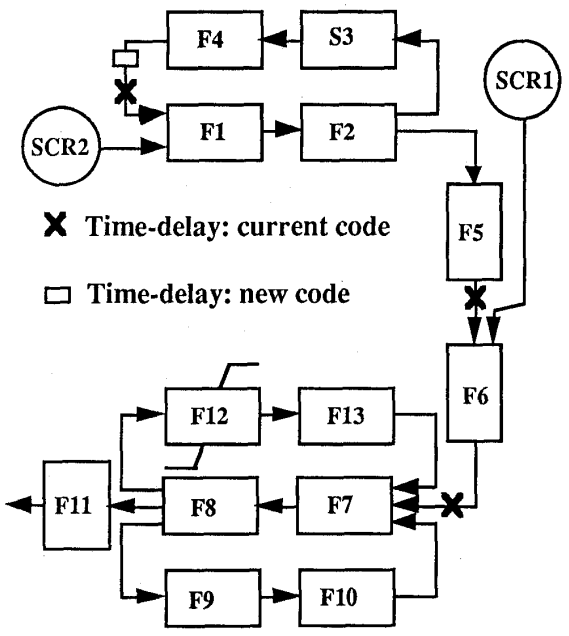

FIGURE 4. Ordering Test system.

In this system the "F" elements are transfer functions and element S3 is a non-linear element. SCR1 and SCR2 are known sources. The ordering system used in V2.0 starts with element F12 to insure correct treatment of the limiter. This would be the correct action if S3 was a linear function. In this case the existing TACS introduced three delays, one between F6 and F7, one between F5 and F6 and the other between F4 and F1. This last delay is the only one required. In a sequential only algorithm three delays are also required. One delay is placed in the F1, F2, S3, F4 loop and two delays are required in the double loop of F7, F8, F9, F10, F12, F13. This last group of elements can be solved in a simultaneous manner. The number of time delays required is now one. This delay must be in the loop containing the non-linear element.

Sequential ordering with super blocks also removes the history term problem by calculating history terms in a sequential manner. For the case of a super block, the full matrix is solved for all internal states before its history terms are calculated. This method avoids the problems of inconsistencies encountered in history terms discussed in Section II.

In the new TACS ordering scheme a matrix is used to describe the relationship between the inputs and outputs of each element in a system. A column and a row of this matrix represents how an element is connected. An element's column has information on which elements provide input signals. Each row of an element has information as to where its outputs are directed. To represent a connection between two elements an "X" will be placed at the intersection of the column representing the element receiving an input and the row representing the element which provides this signal.

The initial form of the ordering matrix for the system in Figure 4 is shown in Figure 5(a). For example, the column for F7 in Figure 5(a) indicates input from F6, F10, and F13. The row for F7 indicates that the output of element F7 is an input to F8. If an element is ordered such that the element(s) providing input precedes the element using the input(s) an "X" is placed in the upper triangle region. If this order is reversed the intersection is in the lower triangle region. For the arbitrary sequence shown in Figure 5(a) there are seven time delay errors indicated by the seven marks in the lower triangle. For example F6 precedes F5. This results in F6 using past output date from F5. The objective of any ordering scheme is to minimize the number of intersections in the low triangle. The first step in the new algorithm is to move all "source-like" elements to the top-left of the matrix. "Source-like" is defined as elements that have inputs from sources or "source-like" elements. The second step is to move all "output-like" elements to the bottom-right of the matrix. "Output-like" is defined as all elements whose output is not used by any element or used only by "output-like" elements. If a control system does not contain loops, the ordering would be complete. For this example SCR1 and SCR2 are in the input region while element F11 is in the output region. These elements are fixed in the matrix and will be neglected in the ordering of the "loop" element. In Figure 5(a) the region to be ordered is the sub-matrix between F6 and F13.

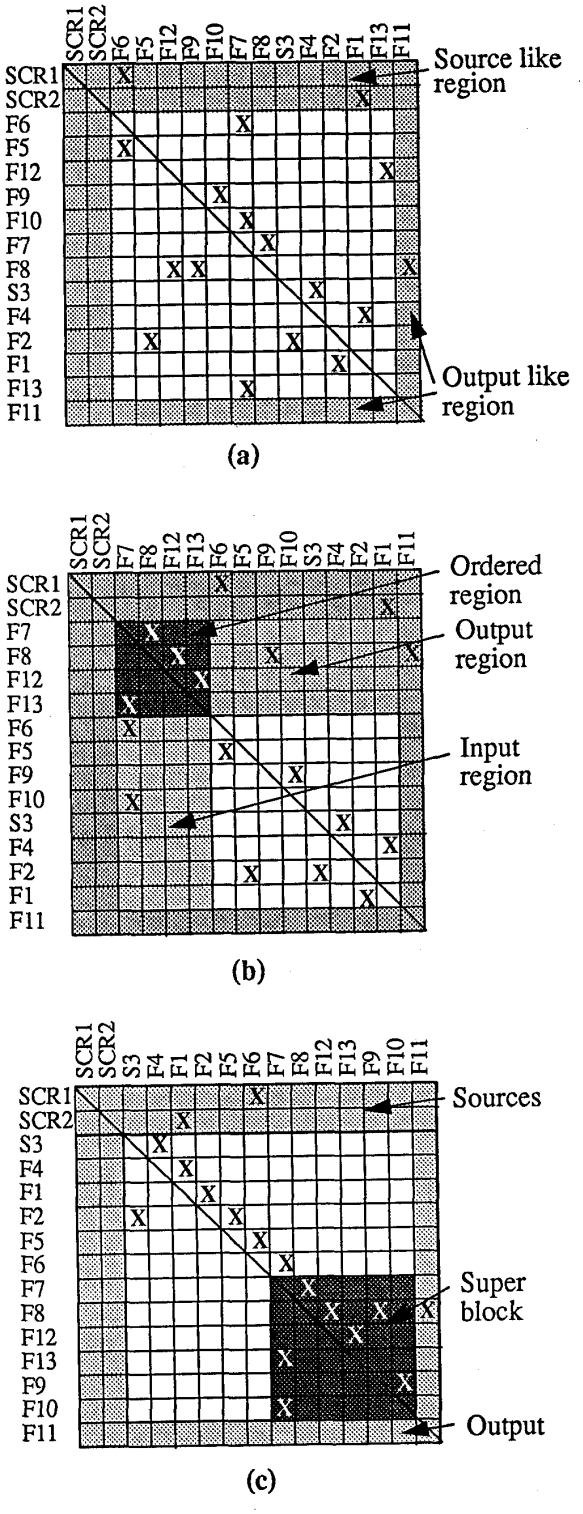

FIGURE 5. Ordering Matrix

The first ordering operation is to find a column with the maximum inputs and move this element to the top of the region. In our example this is element F7. This element is now excluded from the regions to be ordered. The next operation is to look at the outputs of the last element moved, F7 and move this element to follow F7. In our example this is F8.

Figure 5(b) shows the matrix after four forward ordering operations. The next operation is to move an element in the "output" region to a point after F13. In this case this is element F9. This continues until there are no intersections in the "output region." This implies that a loop in the control system has been ordered and there are no additional elements that are part of this loop(s).

If there are no unordered elements left, the ordering is finished. If there are elements left a backward ordering procedure is invoked. In backward ordering an element in the input region defined by the row is moved to the front of the ordered region. This continues until there are no more elements to be ordered. The maximum number of ordering steps are equal to the number of elements. The matrix after the last step is shown in Figure 5(c). Note that there are three intersections in the lower triangle region. This is the best possible solution for sequential ordering. One time step delay for each control loop.

The ordered matrix has important information. First, it provides sequential ordering with a minimum of time delays. Second, it provides information on where the time delays are placed and allows TACS to provided this information to the user. Third, the matrix allows for the identification of loops.

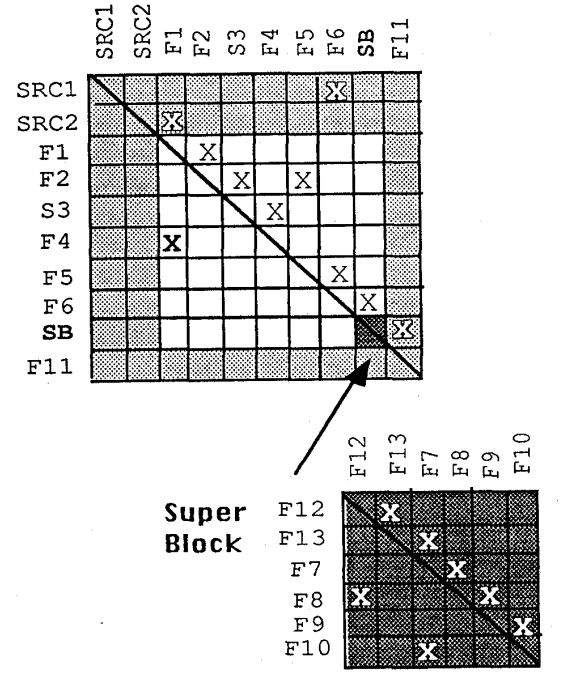

FIGURE 6. Final Form of Ordering Matrix

Elements contained in a square, with its lower left corner being an intersection point for an element in the low triangle, define a loop. For example, in Figure 5(c), F2/S3 intersection contain S3, F4, F1, and F2 comprising a control loop. The second set shown, F10/F7, contains nested loops.

Loops consisting of function blocks only with a single limiter or less can be solved simultaneously using the current methods discussed in Section II. In these cases the algorithm identifies this group of elements as a "super block" and shrinks it to a single element in the ordering matrix and reorders the system. The final ordered form is shown in Figure 6. This contains a single super block for the double loops that contain only transfer functions. The final ordering places a single time delay between elements F4 and F1, see Figure 4. Note that the super block is ordered with the limiter in the first position.

IV. Super Blocks and FORTRAN Models

The super block structure in the ordering procedure allows for the creation of other special features in TACS. In particular the enhanced TACS has two new features that are treated as "super blocks." One feature is an FORTRAN like "IF THEN ELSE" control structure in TACS. A second feature is the ability of TACS to use FORTRAN subroutines as TACS elements.

The current TACS provide many of the intrinsic functions found in FORTRAN. This allows for the creation of FORTRAN like expressions without the need to compile and link new code to the EMTP. The major shortcoming of this modeling tool is its lack of control functions to enable looping, and conditional branching. There are no GOTO, IF, DO, or CASE statements in the current TACS. The addition of an "IF THEN ELSE" structure greatly expands the class of modeling tools available to TACS

The "IF THEN ELSE" structure allows the user to create FORTRAN like models without the problems related to the use of FORTRAN code. The basic structure of this element is as follows:

```
IF (expression) THEN
  VAR1      = expression
  VAR2      = expression
etc.
ELSE
  VAR1      = expression
  VAR2      = expression
etc.
ENDIF
```

The TACS modeling data between the line starting with "IF" and the line containing "ENDIF" is considered a single super block by TACS for ordering. The "expression" is any FORTRAN-like logic or algebraic expression allowed in TACS. The variables VAR1, VAR2, etc., are names associated with the respective FORTRAN-like expressions. The variable names following the "THEN" or "ELSE" section can be unique or shared. These variables can also be internal or external to the block. The algorithm will find which variables are external and which are internal to the "IF THEN ELSE" block. In forming the super block a set of variables will be identified as input variables and another set will be identified as output variables. This block becomes as single element in the Ordering Matrix.

It is also possible to have loops within the block. In TACS the elements within the "IF THEN ELSE" block are assumed to be an independent system with inputs, outputs and internal loops that must be ordered to insure minimum number of time delays. To order this subsystem the identified input variables are handled as "sources" while the output variables are assumed to be "outputs." The remaining components are ordered using the general methods described in Section III.

The enhanced TACS can also use user written FORTRAN code as a TACS device. In this case TACS considers this user supplied device as a super block with multi-inputs and multi-outputs. The inputs and outputs of this user defined device are all TACS variables. They may connect to any other TACS component as required by the control model. Special interface routines are provided to insure correct transfer of data between the user provided FORTRAN subroutine and TACS. This user developed TACS element must also be compiled and linked to the EMTP.

V. Initialization

To reduce simulation time it is important that TACS models be initialized at a steady state operation point. This requires that all variables and related history terms are known. Theoretically TACS can not guarantee correct initialization for an arbitrary control system, but it is possible to greatly improve the current initialization system.

In the current TACS, the initialization algorithm assumes a TACS model that uses only transfer functions. It treats the dc and ac initialization separately. In each case the full system is solved simultaneously using the same methods described in Section II. Upon completing the separate calculations the initial values and history terms for the system are constructed.

For the dc steady state solution the complex frequencies, s, of the transfer functions are set to zero. The resulting matrix is used to find the dc components. For ac initialization the complex frequency, s, is set to $j\omega$. This results in an input-output matrix of complex functions. Again the necessary matrixes are set up and solved for the ac variables and history terms.

Systems modeled solely with sinusoidal sources and transfer functions can be correctly initialized in the current TACS with correction of some minor errors. Unfortunately most control systems have components that are non-linear. Non-linear elements disrupt this initialization method. The current TACS code assumes all nonlinear elements have zero output. For

most TACS models this creates a situation where the initialization is not helpful. The enhanced TACS code removes current errors and provides initial values for some non-linear elements.

The principal initialization error in TACS is related to integrators, $G(s)$ = K/s. Current versions of TACS can find dc steady-state values for all elements except integrators. In this case the dc output is not defined and must be provided by the user. If the user does not provide an output a default sets the output to zero. This is not always the best tactic. In cases where the integrator has an ac input this default strategy is wrong. If the input to the integrator is $\sin(\omega t)$ the expected ac output is $-\cos(\omega t)/\omega$. If a zero output is imposed by default the output then has a dc offset, $[1-\cos(\omega t)]/\omega$. This default problem is resolved by forcing the dc component of the output to zero rather than the actual output.

The general problem of automatic initialization for control loops with non-linear elements is not theoretically solvable. There are many examples where there are no steady state solution for either an ac or dc input. Consider the system shown in Figure 7.

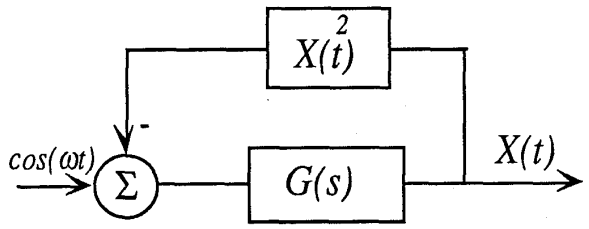

FIGURE 7. Control loop with non-linear element.

This system is a transfer function with a negative feed back being the square of the output. This system does not have a steady state solution for a general transfer function. Even when $G(s) = K$ there are no well defined ac steady solutions. The enhanced TACS takes advantage of the new solution method to provide some help but will not attempt to initialize any loops that contain non-linear elements.

The basic sequential nature of the new ordering methods coupled with the concept of super blocks allow the retention of the best features of the current methods with improvements provided for systems using non-linear elements. In the enhanced TACS any transfer function based super blocks are initialized using current methods.

Any non-linear element that has a defined ac and/or dc output for an ac or dc input can be initialized. For example a function that multiplies two inputs or squares the input will result in an ac output containing two frequencies which are different from that of the inputs. In the case of the enhanced TACS initialization is provided for all elements that have a finite number of ac/dc output components for a single ac/dc input. The outputs of any non-linear elements that have no defined ac or dc component or those which have an infinite number of ac components are set to zero.

The FORTRAN-like non-linear elements, *, -, + and **2 in TACS are initialized in the enhanced code. Some possible functions are given below;

$U_1(t) \cdot K$ $\quad U_1(t) + U_2(t)$ $\quad U_1(t) - U_2(t)$

$U_1(t) \cdot U_2(t)$ $\quad U_1(t)^2$ $\quad \dfrac{U_1(t)}{U_{dc}}$

where $U_1(t)$ and $U_2(t)$ can be an ac or dc signal.

The algorithm is basically sequential and follows the ordering discussed earlier. Each non-linear element has both its dc and ac initial values calculated before the next element in the sequence. This allows for more that one ac frequency. For example, the operation $U_1(t) \cdot U_2(t)$, where $U_1(t)$ and $U_2(t)$ are sinusoidal functions of different frequencies has a well-defined output. The steady state solution is based on the sum and difference of the input frequencies. Since this operation has a definable steady response its initialization is included in the enhanced TACS. Consider another TACS function $\log(U_1(t))$ where $U_1(t)$ is its input. If this input is sinusoidal there is no defined ac steady state solution and TACS can not be expected to initialize this function. This algorithm through Fourier expansion also finds the initial conditions for a periodic input such as a square wave.

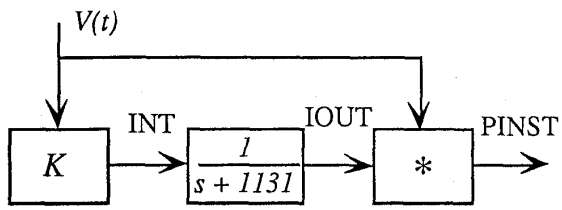

FIGURE 8. Initialization test case

These changes in TACS initialization can be best illustrated through an example, see Figure 8. In this example the input is an ac voltage. This input is scaled by a gain K and becomes the input, INT, to the transfer function. In this example INT is a cosine function with a frequency of 60 Hz and a magnitude of 1.2 Kv. The expected IOUT can be calculated.

$$IOUT(t) = V\cos(\omega t - \theta)$$

where

$$V = \frac{1.2Kv}{\sqrt{337^2 + 1131^2}} = 1.006 volts$$

$$\theta = \text{atan}\left(\frac{337}{1131}\right) = 18°$$

In the current TACS the input to the transfer function would be zero since the multiplication operation is assumed to be non-linear. This zero input would result in the transfer function being initialized to zero. This is seen in the form of the IOUT

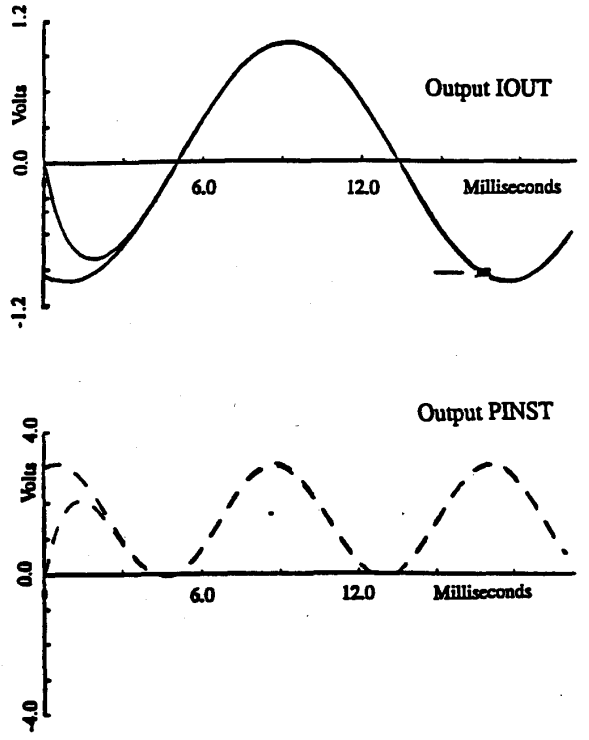

FIGURE 9. Initialization example

curve, Figure 9, which starts at zero and converges to the correct steady state in about 4 milliseconds. This convergence depends on the time constants of the transfer functions. The enhanced code has found the correct initial condition for the transfer function as shown.

The next function in the sequence multiplies two 60 Hz ac wave forms. The theoretical results consist of a dc component and an ac component at twice the input frequency. The comparison is shown in the two curves for PINST. Again the incorrectly initialized output converges to the correct value in 4 milliseconds. In this example the time for the transients to come to the correct steady state value is not large. It is also apparent that different transfer functions could result in a system that would require seconds rather than milliseconds to achieve steady state when not correctly initialized. In this case the new code would greatly reduce the time to achieve steady state. Except for such model related differences the cpu requirements of the old and new code are the same.

VI. Conclusions.

The new enhancements to TACS solve many problems that have plagued users for many years. First, the extra time delays and errors in history term calculations have been removed. Second, the location of all time delays introduced by TACS are provided in the output data for use by the user. Third, the debate over adding new devices to TACS have been squelched by the ability of TACS to use user written FORTRAN code as a TACS element. This allows libraries of special devices to be created and shared between users. Forth, the initialization has been greatly improved. Users should now find initialization useful for most TACS models.

Acknowledgment

The authors wish to acknowledge the support of this work by the EMTP Development Coordination Group. Without there support and guidance this work would not have been undertaken.

References

[1] H.W. Dommel, "Digital Computer Solution of Electromagnetic Transients in Single and Multiphase Networks," *IEEE Transactions on Power Systems*, vol. PAS-88, April 1969, pp. 388-399.

[2] L. Dube and H.W. Dommel, "Simulation of Control Systems in an Electromagnetic Transients Program with TACS," *IEEE Power Industry Computer Application Conference*, 1977, pp. 266-271.

[3] L. Dube, I. Bonfanti, *"MODELS: A New Simulation Tool in the EMTP,"* European Transactions on Electrical Power Engineering, Vol. 2, No.1, pp. 45-50, Jan./Feb. 1992

[4] L.X.Bui, S.Casoria, G.Morin and J. Reeve, *"EMTP TACS-Fortran Interface Development for digital Control Modeling,"* IEEE Trans. PAS, Vol 7, No.1, pp. 314-319, Feb. 1992.

Biographies

Robert H. Lasseter (F'92) received the Ph.D. degree in physics at the University of Pennsylvania, Philadelphia, in 1971.He was a Consultant Engineer at General Electric Company until he jointed the University of Wisconsin-Madison in 1980. His main interests are the application of power electronics to utility systems and simulation methods.

Jia-Rong Zhou is a native of Kunming, Yunnan of the People's Republic of China. He graduated from Chongqing University in Electrical Engineering in 1968. Upon graduation he worked for the Yunnan Research Institute of Scientific and Technical Information in power electronics and later as a computer programer. In 1985 he came to the University of Wisconsin as a visiting scholar. He stayed on to receive his M.S. degree from the University of Wisconsin in 1990. Currently he is with Oak Technologies Inc., Sunnyvale California.