TRANSIENT STABILITY SIMULATION BY WAVEFORM RELAXATION METHODS

M. ILIC'-SPONG
Member, IEEE

M. L. CROW
Student Member, IEEE

M. A. PAI
Fellow, IEEE

Department of Electrical and Computer Engineering
University of Illinois

Abstract

In this paper, a new methodology for power system dynamic response calculations is presented. The technique known as the waveform relaxation has been extensively used in transient analysis of VLSI circuits and it can take advantage of new architectures in computer systems such as parallel processors. The application in this paper is limited to swing equations of a large power system. Computational results are presented.

I. INTRODUCTION

The main trust of this paper is to introduce the waveform relaxation method (WRM) for the transient stability analysis of very large scale power systems. In recent years this method has been shown to be very effective for the transient analysis of VLSI circuits. Although the VLSI systems are technologically newer compared to electric power systems, they share many commonalities with them: the number of nodes typically exceeds several thousand on realistic systems and both circuits are sparse. Also, a typical VLSI circuit is a circuit containing mainly R-C components, while a typical power system consists primarily of R-L components (with some capacitive effects on long lines and as shunt compensation). Because of the dualism between the RC and RL circuits, it is reasonable to assume that the WR techniques can be applied to power systems as well. The implementation of WR algorithms on pipeline or parallel processors will result in enhanced computational efficiency. Hence, the results of this paper complement the recent research work [1]-[3] on applying parallel processors for solving power system problems. Both VLSI circuits and very large scale power systems (VLSP) share the common property that the system matrix is diagonally dominant, if the coupling between machines or groups of machines is weak. In the WR algorithm, this helps in speeding up the convergence. We emphasize here some of the common problems in the VLSP and VLSI systems since the intensive research efforts in the VLSI area can benefit power systems research and vice versa. Moreover, the developments in the VLSI research area are most often geared towards the newest computer technology and architectures [6]. These are important factors in on-line computer implementation or developing software for power system simulators. It is a well known fact that researchers in VLSI simulation have benefited immensely from the pioneering work on sparse matrix techniques first applied to power systems [4]. It is appropriate to emphasize such cross fertilization of ideas in terms of dynamical simulation as a healthy trend.

In particular, in this paper, we present recent results of using waveform relaxation in VLSI circuits [5]-[7] to the transient stability simulation problem with the classical model. The results are general enough to be applicable to multi-machine systems with mixed algebraic and differential equations. Use of parallel processing architectures and numerical convergence aspects are discussed. The simulation results on three, ten, and twenty-machine systems are presented. They are very encouraging and support the viewpoint that new avenues for finding more efficient numerical methods for stability transient analysis (TSA) still exist to the point that the transient stability analysis, if combined with a proper computer technology, may become feasible for real-time monitoring and security evaluation of electric power systems.

II. THE DYNAMICAL MODEL FOR ELECTRIC POWER SYSTEMS

Generally, a multi-machine dynamical model is described by the set of differential and algebraic equations of the type

$$\dot{x} = F(x,y) \ , \ x(0) = x_0 \tag{2.1a}$$

$$0 = G(x,y) \tag{2.1b}$$

where (2.1a) are equations of the generating unit and (2.1b) are those corresponding to the interface equations and the network equations.

If a classical model is assumed for the machine and all loads are converted into impedances, we have equations of the form

$$\dot{X} = F(X) \qquad X(0) = X_0 \tag{2.2}$$

Swing equations with the classical model can be put in the form (2.2) as shown below.

The swing equation for i^{th} machine is

$$M_i \frac{d^2\delta_i}{dt^2} = P_{mi} - E_i^2 G_{ii} - \sum_{\substack{j=1 \\ \neq i}}^{n} (C_{ij} \sin\delta_{ij} + D_{ij} \cos \delta_{ij})$$

$$= P_i - P_{ei} (\delta_1 \ \dots \ \delta_n) \tag{2.3}$$

where

86 SM 331-3 A paper recommended and approved by the IEEE Power System Engineering Committee of the IEEE Power Engineering Society for presentation at the IEEE/PES 1986 Summer Meeting, Mexico City, Mexico, July 20 - 25, 1986. Manuscript submitted February 3, 1986; made available for printing May 7, 1986.

Printed in the U.S.A.

0885-8950/87/1100-0943$01.00©1987 IEEE

$M_i = \frac{H_i}{\pi f}$ with H_i = inertia constant in secs.

P_{mi} = mechanical input in p.u., $P_i = P_{mi} - E_i^2 G_{ii}$

$E_i/\underline{\delta_i}$ = voltage behind the transient reactance X'_{di}

$C_{ij} = E_i E_j B_{ij}$; $D_{ij} = E_i E_j G_{ij}$

G_{ij}, B_{ij} are elements of the Y matrix at the internal nodes of the generator. Depending on the system configuration (faulted or post-fault), these will assume appropriate values.

We take δ_i as the rotor angle with respect to the synchronously rotating reference axis. Let $\dot{\delta}_i = \frac{d\theta_i}{dt} - \omega_s \triangleq \omega_i$ be the relative angular velocity where θ_i is the rotor angle w.r.t. absolute reference frame and ω_s = synchronous velocity (377 rad/sec). Introducing the state variables δ_i and ω_i (i = 1, 2, ..., n), we get the state space equations

$$\dot{\delta}_i = \omega_i \qquad (2.4)$$

$$\dot{\omega}_i = \frac{1}{M_i}\left(P_i - P_{ei}(\delta_1, \ldots, \delta_n)\right) \qquad i = 1, 2, \ldots, n$$

The initial conditions of (2.4) are $(\delta_i(0), 0)$ i = 1, 2, ..., n where $\delta_i(0)$ is computed from the load flow data and x'_{di} of the machine. The usual assumptions regarding P_{mi} = constant and E_i = constant are made during the simulations.

III. BACKGROUND ON THE WAVEFORM RELAXATION METHOD

General Overview

The dynamical model used for transient stability studies of an electric power system consisting of n machines reviewed in Section II, can be thought of as a system of 2n coupled first-order differential equations. If the unknown variables are denoted in a vector form

$$X = [\delta_1\ \omega_1 \quad \delta_2\ \omega_2 \quad \cdots \quad \delta_n\ \omega_n]^T$$

$$= [X_{11}\ X_{12} \quad X_{21}\ X_{22} \cdots \quad X_{n1}\ X_{n2}]^T \qquad (3.1)$$

the power systems equations defined in Section II could be represented as

$$\dot{X}_{11} = F_{11}(X_{11}, X_{12}, X_{21}, X_{22}, \ldots, X_{n1}, X_{n2}) \qquad (3.2a)$$

$$\dot{X}_{12} = F_{12}(X_{11}, X_{12}, X_{21}, X_{22}, \ldots, X_{n1}, X_{n2}) \qquad (3.2b)$$

$$\vdots$$

$$\dot{X}_{n1} = F_{n1}(X_{11}, X_{12}, \ldots, X_{n1}, X_{n2}) \qquad (3.2c)$$

$$\dot{X}_{n2} = F_{n2}(X_{11}, X_{12}, \ldots, X_{n1}, X_{n2}) \qquad (3.2d)$$

Mathematically, the transient analysis problem is a problem of integrating for given initial conditions $X(t_0) = X_0$ equations (3.2a)-(3.2d) in time to obtain X(t) for $t > t_0$ and $t \in [t_0, T]$. The time T is fixed prior to performing the integration and is generally 1-2 seconds. The efficiency of the transient analysis method is mainly measured by the necessary CPU time to generate X(t) for all $t \in [t_0, T]$. Both implicit and explicit methods of integration are used in power system dynamic response calculations [8]. Unique properties of power systems like highly localized fault propagation phenomena or coherency of machines, etc., are not accounted for in the standard transient stability programs, unless coherent dynamic equivalents are established by a separate program. Research in coherency and dynamic equivalents has resulted in good production grade codes [9]-[10]. It is, however, highly desirable that results in coherency be integrated with numerical algorithms and WR algorithm precisely allows for this. Also if some variables do not change during simulation, i.e., latency phenomena, it could be accommodated in our methodology [12].

For any high order problems, direct numerical integration (explicit) methods (DNI) are more attractive than the class of so-called implicit numerical integration (INI) schemes (backward Euler formula being one of the simplest numerically stable in this group). A basic difference between these two methods is that the INI method involves the inversion of a matrix whose order is the same as the order of differential equations to be solved, e.g., 2n for the system (3.2). This is thought of as being computationally expensive for transient analysis studies since the matrix inversion needs to be done at each step in time. Algebraization of the differential equations with LU factorization with sparsity and vector array processors reduce the computation [13]. Typically, if the integration time is $[t_0, T]$ and the integration step is h, the number of inversions would be of order $\frac{[T - t_0]}{h}$, which is extremely time consuming. This reasoning should not be confused with the reasoning in load flow computations, where algebraic, rather than differential equations are solved. Here, Newton-Raphson's method with sparsity and vector array processor reduces the equations of the form AX = b is the most favored approach. In power systems transient analysis, DNI methods like Runge-Kutta are well accepted and used. The proposed Waveform Relaxation Algorithms as a new option are originally based on INI methods (like Backward Euler Formula) and then combined with the latency [12] property of diagonally dominant dynamical systems, to reduce the order of matrices which need to be inverted.

Waveform Relaxation Method

The Waveform Relaxation Algorithms are illustrated first on a simple system of two differential equations in two unknowns, X_1, X_2:

$$\dot{X}_1 = F_1(X_1, X_2) \quad X_1(t_0) = X_{10} \qquad (3.3a)$$

$$\dot{X}_2 = F_2(X_1, X_2) \quad X_2(t_0) = X_{20} \qquad (3.3b)$$

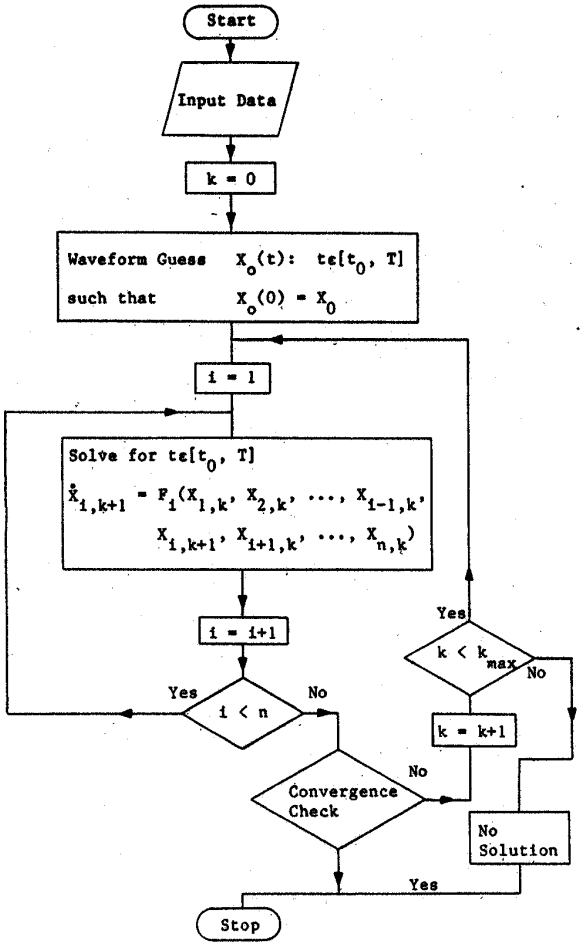
The basic idea of the WRM is to fix the waveformn X_2 in $[t_0, T]$ and integrate equation (3.3a) as a one-dimensional differential equation in $X_1(t)$ over the whole time interval $[t_0, T]$ ("one sweep"). The solu-

tion obtained for $X_1(t)$ can be substituted in (3.3b), which will then reduce to another first-order differential equation in one variable $X_2(t)$. Equation (3.3a) is then integrated again using the new solution $X_2(t)$ and the procedure becomes iterative. A standard numerical integration problem of n differential equations in n unknowns becomes a problem of solving iteratively a sequence of n differential equations in one variable. This algorithm is analogous to the Jacobi method for solving the load flow equations. Details can be found in [14].

The basic WRM for solving a general system of nonlinear differential equations (2.2) is presented in Flow Chart 1.

FLOW CHART 1.

The Basic Waveform Relaxation Method.



k_{max} = maximum number of sweeps

A typical convergence check is

$$\max_{1\le i\le n} \max_{t\varepsilon[t_0, T]} |X_{i,k+1}(t) - X_{i,k}(t)| \le \varepsilon \qquad (3.4)$$

Important specific features of this numerical algorithm compared to conventionally used power systems transient stability schemes are:

(a) the algorithm is iterative

(b) the step where the solution in the $(k+1)^{st}$ sweep of differential equations is employed involves solving a differential equation in one unknown $X_{i,k+1}$ only. The other variables $X_{1,k}, X_{2,k}, \cdots, X_{i-1,k}, X_{i+1,k}, \cdots, X_{n,k}$ are known from the previous sweep k. In the VLSI literature, a typical INI method used to perform this step is the backward Euler algorithm.

We adopt the same method here, and it basically amounts to the following. Letting j correspond to the integration step i, to the ith state variable and k to the sweep

$$\dot{X}^{j}_{i,k+1} = F_i\,(X^{j}_{1,k}, X^{j}_{2,k}, \cdots, X^{j}_{i-1,k}, X^{j}_{i,k+1}, X^{j}_{i+1,k}, \cdots, X^{j}_{n,k}) \qquad (3.5)$$

or

$$\frac{X^{j+1}_{i,k+1} - X^{j}_{i,k+1}}{h} = F_i\,(X^{j}_{1,k}, X^{j}_{2,k}, \cdots, X^{j}_{i-1,k}, X^{j}_{i,k+1}, X^{j}_{i+1,k}, \cdots, X^{j}_{n,k}) + \sum_{\substack{\ell=1 \\ \ell\neq i}}^{n} \frac{\partial F_i}{\partial X_\ell}(X^{j}_{1,k}, \cdots, X^{j}_{i,k+1}, \cdots, X^{j}_{n,k})(X^{j+1}_{\ell,k} - X^{j}_{\ell,k}) + \frac{\partial F_i}{\partial X_i}(X^{j}_{1,k}, \cdots, X^{j}_{i,k+1}, \cdots, X^{j}_{n,k})(X^{j+1}_{i,k+1} - X^{j}_{i,k+1}) \qquad (3.6)$$

which implicitly defines

$$X^{j+1}_{i,k+1} = X^{j}_{i,k+1} + h\Big(1 - h\frac{\partial F_i}{\partial X_i}(X^{j}_{1,k}, \cdots, X^{j}_{i,k+1}, \cdots, X^{j}_{n,k})\Big)^{-1} \cdot \Big(F_i\,(X^{j}_{1,k}, \cdots, X^{j}_{i,k+1}, \cdots, X^{j}_{n,k}) + \sum_{\substack{\ell=1 \\ \ell\neq i}}^{n} \frac{\partial F_i}{\partial X_\ell}(X^{j}_{1,k}, \cdots, X^{j}_{i,k+1}, \cdots, X^{j}_{n,k})(X^{j+1}_{\ell,k} - X^{j}_{\ell,k})\Big) \qquad (3.7)$$

If higher order terms in the right-hand side of (3.7) are neglected, (3.7) can be approximated as

$$X^{j+1}_{i,k+1} = X^{j}_{i,k+1} + h\Big(1 - h\frac{\partial F_i}{\partial X_i}(X^{j}_{1,k}, \cdots, X^{j}_{i,k+1}, \cdots, X^{j}_{n,k})\Big)^{-1} \cdot F_i\,(X^{j}_{1,k}, \cdots, X^{j}_{i,k+1}, X^{j}_{n,k}) \qquad (3.8)$$

IV. CONVERGENCE CONDITIONS AND PROCEDURES FOR SPEED-UP

It has been shown that rather mild conditions on continuity of functions F_i, i = 1, ..., n are required for the WRM to converge for any initial guess $X(t_0)$, $t\varepsilon[t_0, T]$. (Theorems 3.1, 3.1 in [5]).

For typical power systems the required conditions are always satisfied. Going back to the parallelism with the Jacobi method for load flow solution, a well known, theoretically proven fact [23] is that this method converges for a wider range of initial guesses than the standardly used Newton-Raphson method. It is the rate of convergence that makes the Jacobi method inferior for load flow calculations. Jacobi method is considered to converge linearly, while Newton-Raphson converges almost quadratically. Since the WRM technique in transient analysis is analogous to Jacobi's method for solving algebraic equations whose convergence rate is linear, this might be a point of concern. Gauss-Seidel WRM algorithm will improve the convergence rate [5].

It is not surprising, however, that the initial convergence rate obtained for VLSP systems is much higher than the convergence rate for VLSI systems. We argue in the next section that if the knowledge about coupling between machines on a typical large scale power system is used, the WRM easily becomes a superior technique, even prior to parallel processing implementation.

A true issue in comparing convergence rate of conventionally used direct integration methods for transient analysis with the WRM is the fact that the DNI methods become inefficient for extremely large problems because of the following two reasons.

(a) The sparse matrix solution time grows superlinearly with the size of the problem. Experimental evidence indicates that the point where the matrix solution time begins to dominate is when the system has over several thousand nodes, which is the case for VLSP systems.

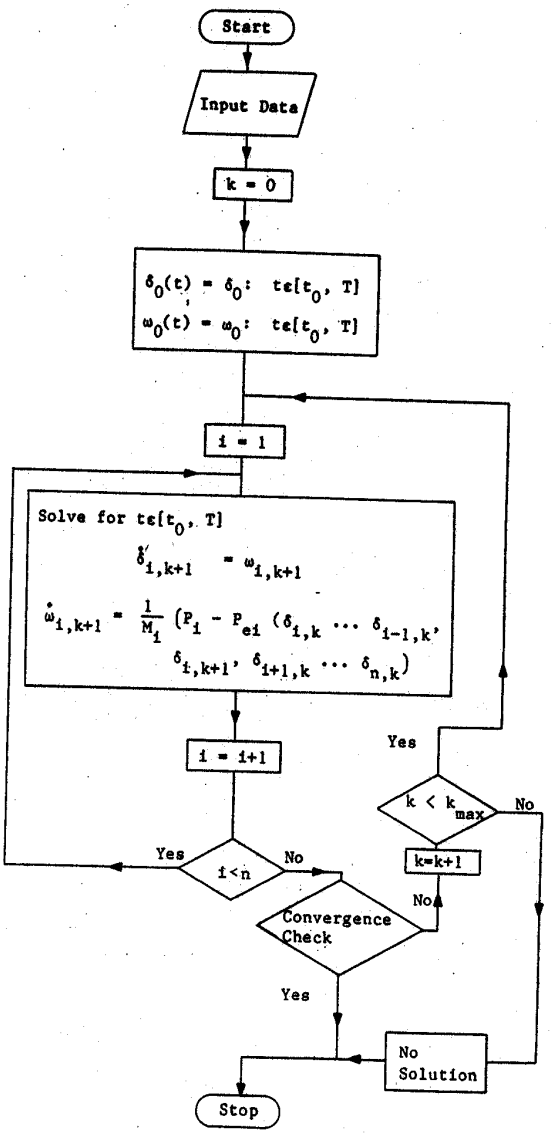
(b) The DNI methods become inefficient for large problems when the differential equations are stiff. Direct application of the integration method forces every differential equation in the system to be discretized identically, and this discretization must be small enough so that the fastest changing variable in the system is accurately represented. Gear's formula [16] overcomes this difficulty. If it were possible to pick different discretization points, or time steps, for groups of differential equations in the system so that each could use the largest time step that would accurately reflect the behavior of the associated variables, the efficiency of the simulation would be greatly improved. This is the multiple time-scale issue and has been implemented in a DNI framework [17].

Specific Techniques Employed

The partitioning of original power equations (2.4) is done in 2 x 2 blocks, i.e., the angle δ_i and speed ω_i are solved from two equations simultaneously. The notation corresponding to the power system problem is given in (3.1). If equations are solved one at a time, the WRM does not converge. So, the results reported here are for the case where two equations are solved in two unknowns by the Backward Euler Formula and the WRM adopted to this simulation is given by Flow Chart No. 2.

FLOW CHART 2.

WRM Algorithm for Transient Stability Analysis



k_{max} = maximum number of sweeps

Different time responses of different variables for which we are solving naturally suggest grouping based on coherency [9]-[11]. Initial experiments with this show that the convergence rate is rather sensitive to grouping. If the machines within a group are tightly coupled, then the convergence rate in terms of number of "sweeps" is better. WR algorithm can thus incorporate coherency based technique with numerical integration. A major research effort on VLSP systems with nodes of order of thousands should be based on exploring this fact together with the WRM.

The other technique for speeding up the WRM convergence rate is based on the so-called time windowing [6]. If we are interested in results on time $[t_0, T]$, the WRM could be applied on $[t_0, T_1]$ then in $[T_1, T_2]$ and $[T_2, T]$ in order to reduce the total CPU time. The effect of time windowing on the numerical examples is presented in the next section.

The grouping of differential equations to be solved simultaneously has a significant effect on very large scale systems and it should be done carefully. The "adaptive clustering" idea reported in the VLSI literature [6] could be directly implemented since grouping under the same name and meaning has been recently reported by Zaborszky et al. [18]. Again, this is what makes this numerical method very promising and currently research is in progress in this area.

V. PARALLEL PROCESSING BASED WRM

An additional property of WRM is the fact that parallel processing is a natural setup for the method. If one processor is assigned to one group of variables which are solved for simultaneously, the CPU time is reduced proportionally to the number of groups. Considerable work exists on parallel solution of ordinary differential equations [19]-[20].

A detailed discussion of the possible computer architectures for WRM methods is given in [6]. Previously reported work on parallel processing in power systems [1]-[3] exploits time-point pipelining algorithms. The shared memory computer architecture is argued to be more efficient for the WRM support of VLSI systems which, we believe, should be common for VLSP systems. The key problem in designing a parallel processor lies in the communication between the processors and is much easier here.

The main advantage of a shared memory system is that it is not necessary to explicitly transfer data from one processor to another. When a processor needs data from another processor, it simply reads from the memory location in which the other processor has written. This allows for more dynamic algorithm structures (like WRM with fault dependent grouping) because it is not necessary to determine beforehand which processors will need the results of a given calculation. There are disadvantages, however (synchronization and locking being the major), but overall the shared memory architecture appears to be better for more adaptive algorithms.

VI. NUMERICAL RESULTS

The WR algorithm discussed in Section III was implemented on the 3, 10 and 20 machine systems. The 3 machine data is taken from Ref. [21], and 10 and 20 machine data from Ref. [22]. From the numerical point of view it appears that the number of sweeps is fairly independent of the size of the system. This is probably due to the fact that the natural modes of the system lie in a fairly small range. In each case we have compared the WR algorithm with the explicit fourth order Runge-Kutta method. Both simulations were coded in PASCAL so that CPU times etc. can be compared on the same basis. The step size chosen was Δt = .00375s because the comparison was with Runge-Kutta method which becomes unstable for larger Δt. If an implicit method such as the trapezoidal method is used, Δt could have been chosen higher. In fact, in the WR algorithm, Δt can be chosen even higher because of the backward Euler method which is numerically stable.

The CPU times were computed in each case and are shown in Table I. Also shown are the number of "sweeps" for the WR algorithm and the CPU time with Δt = 0.0375 s. The progressive convergence for $\delta_6(t)$ in the 10 machine case for different sweeps is shown in Figure 1. There was no significant change in number of sweeps when Δt was increased from .00375 s to .0375 s. It can be seen that CPU time decreases for the WR algorithm when Δt is increased to .0375 s.

	3 m/c	10 m/c	20 m/c
1. CPU time (WR) Δt = .00375s	4.440	36.232	151.941
2. CPU time (WR) Δt = .0375s	0.402	3.072	21.337
3. CPU time (RK) Δt = .00375s	1.130	7.291	26.674
4. No. of Sweeps (WR) Δt = .00375s	19	18	19

TABLE I. CPU time comparison and number of "sweeps" for WR algorithm.

The effect of "windowing" in the WR algorithm is shown in Table II. Two different time "windowing" patterns were investigated. There is a significant decrease in CPU time when windowing is used. Only the results using three "windows" are presented in Table II. Comparing these results with the corresponding cases 1 and 2 in Table I, we observe a 50% reduction in CPU time. Since "windowing" is not possible in the RK method, no direct comparison is possible. At this point, it is felt that depending on the time interval of simulation $[t_0, T]$ beyond a certain number of "windows," one cannot expect a decrease in CPU time. The same experience has been reported for the VLSI systems [6]. Figure 2 shows the effect of "windowing" on $\delta_6(t)$ in the 10 machine case.

	3 m/c	10 m/c	20 m/c
1. CPU time (WR) Δt = .00375s	2.436	17.889	79.053
2. CPU time (WR) Δt = .0375s	0.220	1.614	9.605

TABLE II. CPU time with "windowing" (.5, 1.0, 1.5 S)

Finally, the WR algorithm in combination with coherent grouping was investigated. In the 10 machine case there were six coherent groups as in [10]. The groups were machines (2,3), (4,6,7,8,10), (5), (9), (1) (see Figure 3). Instead of grouping the variables as in (3.1), we group all variables corresponding to a coherent group as one set. If the coupling between the areas is weak and within an area is tight, this will have the effect of improving both the CPU time and the number of sweeps. The improvement is dependent on the degree of coherency. Table III shows the effect of grouping on the 10 and 20 machine system.

	10 m/c	20 m/c
1. CPU time (WR) Δt = .00375s	29.961	105.147
2. CPU time (WR) Δt = .0375s	3.075	14.806
3. CPU time (RK) Δt = .00375s	14	13

TABLE III. Effect of grouping.

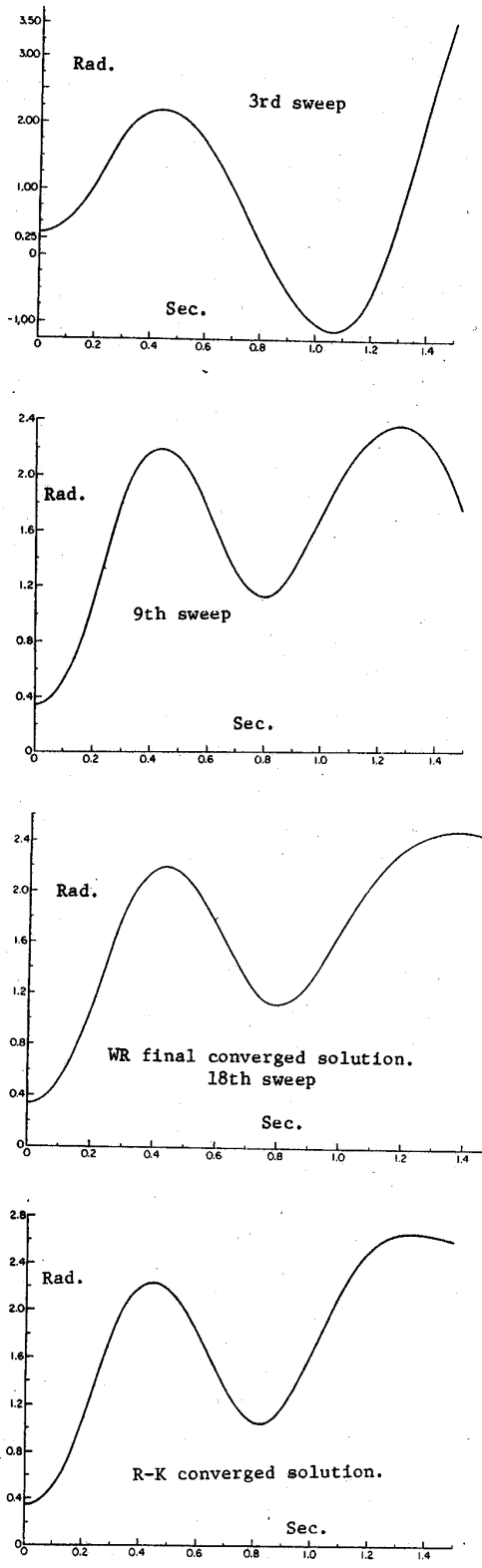


Figure 1. Successive sweeps of $\delta_6(t)$ for fault at bus #35 cleared in .24 Sec. (Δt = .00375 Sec.) and solution by R-K Method

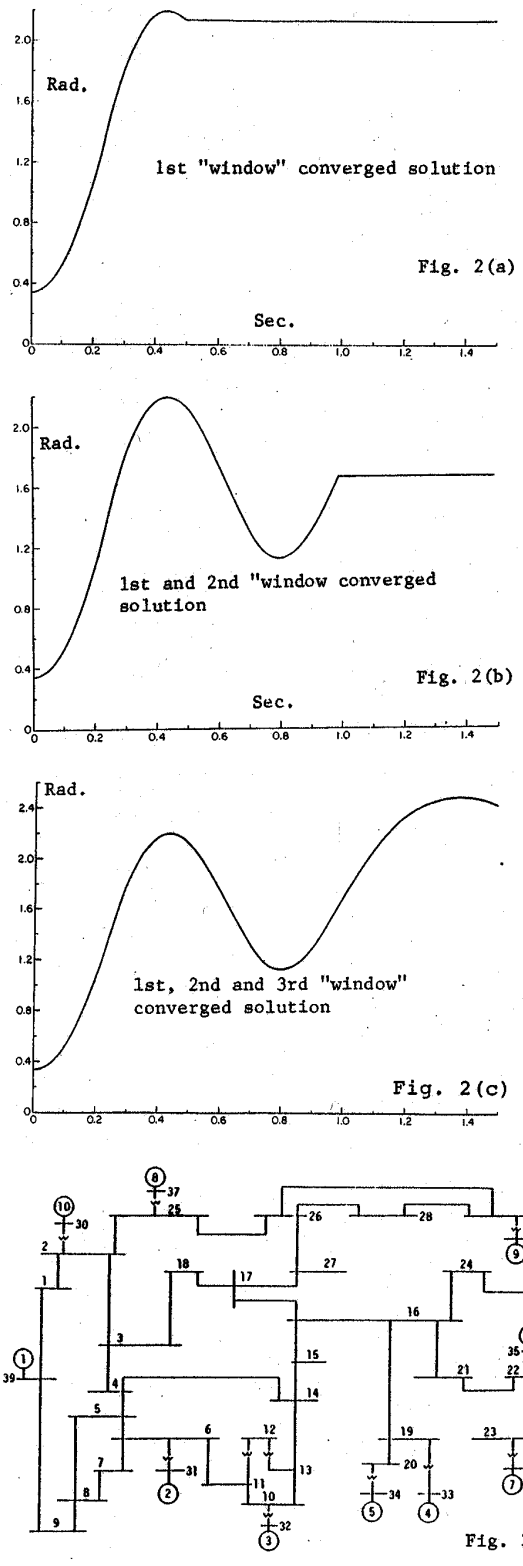


Figure 2. Converged solution by "windowing" technique (.5, 1.00, 1.5 Sec.)

Figure 3. Single line diagram of 10 machine, 39 bus system.

Note tl
still n
Tables
solving
shortes
It is t
the rat
for the
case fc
the poe
in whic
A compa
the tra
the lat
eratior
method
aspect
argumer
cessors
WRM of
(9.605,
like c
determ

VII. !

In thi
analys
posed.
sively
algebr
iterat
flow
Curren
models
make
result
numeri

Acknov

The a
of th
J. R.
autho
Progr
ackno
Scien
autho

REFER

[1]

[2]

[3]

[4]

[5]

Note that a combination of windowing and grouping still needs to be studied. Based on the results in Tables II and III, we believe that the CPU time for solving a 20 machine case will be shorter than the shortest reported for the WRM, i.e., 9.605 CPU. It is true that as the size of the system increases, the ratio of CPU time of WR/RK increases from 2.155 for the three machine case to 2.96 for the 20 machine case for $\Delta t = .00375$ sec. However, this is offset by the possibility of taking larger Δt for the WR method in which case the same ratio varies from .1946 to .36. A comparison of the WR method with a stable method like the trapezoidal or backward Euler method indicated that the latter takes much computer time. Sparsity considerations were not used and hence a direct comparison method is not fair at this stage. Research on this aspect is continuing. If the parallel processing argument is introduced with a suggested number of processors M, the reported computing time needed for the WRM of 9.605 CPU would reduce to approximately r the (9.605/M) CPU. However, many different constraints, like cost and specific computer architecture, should determine the choice of the number of processors M.

VII. CONCLUSION

In this paper a new method for the transient stability analysis of very large scale power systems is proposed. This is the waveform relaxation method intensively used in VLSI systems. Groups of differential-algebraic equations are integrated for the "sweep" and iteratively done as in Jacobi method of solving load flow equations. The technique is very promising. Current further work in this area concerns the detailed models of machines and use of parallel processors to make the tool an effective one. Also, theoretical results on coherency are being combined with the numerical aspects of the WRM.

Acknowledgements

The authors thank Professor P. W. Sauer and I. N. Hajj of the University of Illinois and Drs. J. H. Chow and J. R. Winkelman for their useful discussions. The authors acknowledge the support of the Power Affiliates Program of the University of Illinois. They also acknowledge research support under the National Science Fouccation grants NSF-ECS83-52221 by the first author and NSF-ECS84-14677 by the third author.

REFERENCES

[1] H. H. Happ, C. Pottle and K. A. Wirgau, "Future computer technology for large power system simulation," Automatica, vol. 15, pp. 621-629, 1979, Pergamon Press, U.K.

[2] F. M. Brasch, Jr., P. E. Van Ness and S. C. Kang, "A multiple processor network for power system dynamics problems," in Parallel Processing for Power System Planning and Operations, EPRI Special Report, October 1980.

[3] R. C. Dugan, I. Durham and S. N. Talukdar, "An algorithm for power system simulation by parallel processing," Paper A-79-487-0, IEEE PES Summer Meeting, Vancouver, B.C., Canada, July 1979.

[4] W. F. Tinney and J. W. Walker, "Direct solutions of sparse network equations by optimally ordered triangular factorization," Proc. IEEE, vol. 55, pp. 1801-1809, 1967.

[5] E. LeLarasmee, A. E. Ruehli and A. L. Sangiovanni-Vincentelli, "The waveform relaxation method for time domain analysis of large scale integrated circuits," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. CAD-1, no. 3, pp. 131-143, July 1982.

[6] A. L. Sangiovanni-Vintecelli, "Waveform relaxation techniques and their parallel implementation," Proc. 24th IEEE-CDC Conference, vol. 3, pp. 1544-1551, Ft. Lauderdale, FL, December 1985.

[7] J. White, F. Odeh, A. L. Sangiovanni-Vintecelli and A. Ruehli, "Waveform relaxation--theory and practice," Transactions on Computer Simulation, to appear.

[8] B. Stott, "Power system dynamics response," Proc. IEEE, vol. 67, no. 2, February 1979.

[9] P. V. Kokotovic, B. Avramovic, J. H. Chow and J. R. Winkelman, "Coherency based decomposition and aggregation," Automatica, vol. 18, pp. 47-56, 1982.

[10] R. Podmore, "Identification of coherent generators for dynamic equivalents," IEEE Transactions on Power Apparatus and Systems, vol. PAS-97, pp. 1344-1354, 1978.

[11] R. Podmore, et al., "Development of dynamic equivalents for power system studies," EPRI EL-456 Project 763 Final Report by Systems Control, Inc., May 1977.

[12] N. B. G. Rabbat, A. L. Sangiovanni-Vincentelli and H. Y. Hsieh, "A multi-level Newton algorithm with macro-modelling and latency for large scale circuits in the time domain," IEEE Transactions on Circuits and Systems, vol. CAS-26, pp. 733-741, September 1979.

[13] H. W. Dommel and N. Sato, "Fast transient stability solutions," IEEE Transactions Power Apparatus and Systems, vol. PAS-91, July/August 1972.

[14] G. W. Stagg and A. H. El-Abiad, Computer Methods in Power System Analysis, McGraw-Hill, New York, 1968.

[15] J. M. Ortega and W. C. Rheinboldt, Iterative Solutions of Nonlinear Equations in Several Variables, Academic Press: New York, 1970.

[16] G. W. Gear, Numerical Initial Value Problems of Ordinary Differential Equations, Prentice-Hall: New Jersey, 1971.

[17] M. A. Pai and R. P. Adgaonkar, "Dynamic equivalents of power systems using singular perturbation technique," Proceedings of the IFAC Symposium on "Computer Applications in Large Scale Systems," New Delhi, India, August 1979.

[18] J. Zaborszky, G. Huang and K. W. Liu, "A textured model for computationally efficient reactive power control and management," IEEE Transactions on Power Apparatus and Systems, vol. PAS-104, pp. 1712-1727, July 1985.

[19] M. A. Franklin, "Parallel solution of ordinary differential equations," IEEE Transactions on Computers, vol. 27, pp. 413-420, 1978.

[20] F. L. Alvarado, "Parallel solution of transient problems by trapezoidal integration," IEEE Transactions on Power Apparatus and Systems, vol. PAS-97, 1978.

[21] P. M. Anderson and A. A. Fouad, Power System Control and Stability, Iowa State University Press, Ames, IA, 1977.

[22] T. Athay, et al., "Transient stability analysis," Conference on System Engineering for Power, Switzerland 197 DOE Publication Conf-790904-PL, 1980.

[23] M. Ilić-Spong, et al., "Block diagonal dominance of nonlinear equations with applications to load flow equations in power systems," Journal on Mathematical Modelling, vol. 5, pp. 275-297, 1984.

Discussion

P. Kundur and **G. J. Rogers** (Ontario Hydro, Toronto, ON, Canada): This is a very interesting paper and describes techniques which appear to have much potential in the analysis of large scale power system transient stability problems.

At present we are simulating systems with as many as 10 000 buses and 1000 synchronous machines. With a conventional stability program using either explicit or implicit integration techniques, this requires several hours of CPU time on a computer such as VAX 8600. A method of reducing CPU time while retaining the ability to model very large systems in detail would be well received by the power industry.

The current work of the authors shows how the waveform relaxation (WR) method can be used for implementing an implicit integration technique in a simple and efficient manner without the need for the simultaneous solution of a very large system of algebraic equations. This simplifies programming considerably and removes a major limitation of the implicit integration approach. The authors have shown that coherent grouping of machines for simultaneous integration of parts of the system further improves the speed of solution. This points to the multi-time scale property of power systems as being the physical basis by which the method could be extended to the incorporation of the dynamic effects of synchronous machines and associated controls. Have the authors considered such an extension?

The authors have not commented on storage requirements of the WR method. Unless reasonably small time windows are used, the need for storing all the state variables for every time step of the entire study period would make the method unattractive.

We find it difficult to believe that with a fourth order Runge–Kutta (RK) integration method a time step as small as 0.00375 s had to be used. With all generators represented by classical models as in the paper, we have simulated fairly large systems using time steps as large as 0.1 s with fourth order RK methods. Even when detailed models including the effects of high initial response exciters and PSS are used, we are able to use time steps of the order of 0.02 s. We wonder if the authors used any approximations in the implementation of the RK method. It is also not apparent how coherent grouping was implemented with the RK method. Our experience has been that the RK methods require truly simultaneous solution of all the states at each time step.

We believe that the concepts presented in the paper could be very effectively used in the simulation of the dynamic performance of very large power systems, with computational gains significantly better than those demonstrated in the paper.

Manuscript received August 1, 1986.

Adrián Inda (Instituto de Investigaciones Eléctricas, Cuernavaca, Morelos, México): The authors are to be commended for their interest in finding new methodologies for transient stability calculations. First of all, I would like to make some comments regarding your statement about implicit numerical integration (INI) schemes. In section III of the paper under "General Overview," the authors state that whenever INI is applied, the inversion of a matrix whose order is the same as the order of the differential equations to be solved is needed. This is not necessarily true. A wise arrangement of the differential equations can lead to an appropriate application of the INI without having to invert a large matrix. To better explain this idea, let us recall that the application of an INI to a set of first order ordinary differential equations of the form

$$\underline{f}(\underline{X}, \underline{U}) = \underline{A}\underline{X} + \underline{B}\underline{U} \tag{a}$$

results in

$$|\underline{I} - kh\underline{A}|\underline{X}(t) = kh\underline{B}\underline{U}(t) + \underline{K}(t - \Delta t). \tag{b}$$

For the transient stability problem, some elements of $\underline{A}$ and $\underline{B}$ change due to saturation.

Instead of obtaining $\underline{X}(t)$ from (b) by matrix inversion for the whole system of equations, partitioning can be used as suggested by B. Stott in Ref. [8] of the paper and analytical solutions for each machine of the system (including controls) can be coded directly for any desired number of modeling options. This manner of applying INI methods, besides avoiding the inversion of large matrices, can handle the problem of saturation with no extra effort.

The discusser has applied this idea in developing a new transient stability program and so far no problems have been encountered [a].

I think this partitioning idea can be very useful if properly applied with the WRM proposed by the authors, especially if detailed modeling is to be used for the synchronous machine controls. The authors' opinion on this idea would be very much appreciated.

Another point which I would like to address in this discussion is related to the idea of parallel processing. The authors argue that WRM is naturally suited for parallel processing and I agree with them. However, in my opinion WRM is not the only method which can result naturally suited for parallel processing.

Following the above partitioning idea, a partitioned-implicit solution scheme together with network tearing techniques can result also in a parallel processing naturally suited structure. Suppose you have arranged the differential equations in such a way that matrix $\underline{A}$ of Eq. (a) presents a matrix diagonal structure; $\underline{A} = \text{dig}\,|\underline{A}_1, \underline{A}_2, \cdots, \underline{A}_n|$ and matrix $\underline{B}$ a block rectangular structure.

For the algebraic equations consider the nodal formulation

$$\underline{I}(\underline{X}, \underline{V}) = \underline{Y}_{BUS}\underline{V} \tag{c}$$

If saturation, saliency, and nonlinear loads are considered, the transient stability algorithm consists in solving (b) and (c) for each integration step alternatively within a global iterative process until convergency is achieved.

Now suppose a tearing technique has been wisely applied to the system, leaving it torn in n-subsystems, each one of them with the following structure:

$$\begin{vmatrix} \underline{I}_i \\ \underline{I}_{i_{INT}} \end{vmatrix} = \begin{vmatrix} \underline{Y}_{ii} & \underline{Y}_{i_{INT}} \\ \underline{Y}_{INT_i} & \underline{Y}_{INT} \end{vmatrix} \begin{vmatrix} \underline{V}_i \\ \underline{V}_{i_{INT}} \end{vmatrix} \tag{d}$$

Apply a triangulation process to get:

$$\begin{vmatrix} \underline{I}_i \\ \underline{I}_{i_{INT}} \end{vmatrix} = \begin{vmatrix} \diagdown & \underline{U}_i \\ \underline{O} & \underline{Y}_{INT} \end{vmatrix} \begin{vmatrix} \underline{V}_i \\ \underline{V}_{i_{INT}} \end{vmatrix} \tag{e}$$

Since the $\underline{Y}_{BUS}$ matrix remains constant as long as no switching events occur, the elimination process of (e) has to be performed only when switching operations occur.

The transient stability algorithm with this modeling would be for each time step:

1) Using previous values of $\underline{U}$ estimate an initial value of $\underline{U}(t)$ for each subsystem.
2) Solve differential equations by applying (b) to each subsystem exploiting matrices $\underline{A}$ and $\underline{B}$ block structure.
3) Solve interconnection equation for the subsystems all together, to calculate $\underline{V}_{INT}$.
4) Obtain $\underline{V}_i$ for each subsystem by backward substitution from (e).
5) Check for convergency of the process. If process is converged, go for the next time interval, otherwise go back to step 2) through 4) until convergency is obtained.

The next step, of course, would be to apply this partitioned solution scheme with parallel processing techniques, with each subsystem being solved by a different processor.

I believe this solution algorithm can render low CPU execution time, which could be compared with the authors' idea.

It is worth mentioning that the numerical stability properties of the INI methods are preserved under this partitioned process, since the whole system of equations are taken to convergency.

The authors' comments about applying these ideas to the WRM they propose are to be very welcome.

Once again, I congratulate the authors for such an interesting paper.

References

[a] A. Inda, A. Medina, R. Fraga, and O. Reynaga, "DINAMIC: Un Programa Digital Eficiente para el Análisis de Estabilidad Transitoria en Sistemas de Potencia," Accepted for presentation in MEXICON-86, International Conference IEEE, México Section, Guadalajara, Jal., October, 1986.

Manuscript received August 13, 1986.

O. Razavi, H. D. Chiang, and **P. Varaiya** (University of California, Berkeley, CA): The authors have presented an interesting application of the Waveform Relaxation Method (WRM) to the power systems. We wish to put forward the following comments.

As properly noted by the authors, exploitation of the inherent properties of the power systems could significantly improve the computation time of the WRM-based simulations. Regarding the two properties mentioned by the authors, namely "coherency" and "latency," we would like to inquire

a) what metho
the test cases
simulation exp
In the introd
mixed algebra
algebraic equ
presented here
The WRM
dynamical equ
for the wavefc
During each s
over the time
illustrated as i
each sweep a
actual solutio
However, t
Using the sam
write (3.8) as

$X^{j+1}_{i,l}$

where

At the first ti

$X^{1}_{i,k}$

Notice that t
therefore, it

At the secon

X^{2}_{i}

Notice that b
1, i.e.,

and so on f
algorithm m
the actual sc

Manuscrip

Fi

what method they used to obtain the coherency grouping information for test cases and b) whether they observed latency phenomenon in their ıulation experience and if so, of what nature.

In the introduction section, it is stated that the results may be applicable to xed algebraic and differential equations. We would like to ask how the ebraic equations may be incorporated into the solution framework sented here.

The WRM is an iterative process. It starts by decomposing the set of namical equations (2.3) into disjoint subsystems. Then, an initial guess the waveform solution of the original dynamical equations is provided. ıring each sweep (or iteration), each decomposed subsystem is solved for er the time interval $[t_0, T]$. The solution process for state X_i may be ıstrated as in Fig. (A.1). There is a flat initial guess ($k = 0$) and after ch sweep an improved waveform is obtained until convergence to the tual solution waveform is achieved.

However, the algorithm as presented by (3.8) is found to act differently. ıing the same notation as in the paper, for the sake of simplicity, we may rite (3.8) as follows

$$X^{j+1}_{i,k+1} = X^{j}_{i,k+1} + G_i(X^{j}_{i,k}, \cdots, X^{j}_{i,k+1}, \cdots, X^{j}_{n,k}) \tag{A.1}$$

here

$$G(\cdot) \triangleq h \cdot \left(1 - h\frac{\partial F_i(\cdot)}{\partial X_i}\right)^{-1} \cdot F(\cdot) \tag{A.2}$$

t the first time step, (A.1) becomes:

$$X^{1}_{i,k+1} = X^{0}_{i,k+1} + G_i(X^{0}_{i,k}, \cdots, X^{0}_{i,k+1}, \cdots, X^{0}_{n,k}) \tag{A.3}$$

otice that the right-hand side (rhs) of (A.3) is evaluated at time t_0 and ierefore, it remains constant for all sweeps $k > 0$, i.e.,

$$X^{1}_{i,k+1} = X^{1}_{i,1} \tag{A.4}$$

t the second time step, (A.1) becomes:

$$X^{2}_{i,k+1} = X^{1}_{i,k+1} + G_i(X^{1}_{i,k}, \cdots, X^{1}_{i,k}, \cdots, X^{1}_{n,k}) \tag{A.5}$$

lotice that because of (A.4), the rhs of (A.5) is constant for all sweeps $k >$, i.e.,

$$X^{2}_{i,k+1} = X^{2}_{i,2} \tag{A.6}$$

nd so on for the rest of the time points. Therefore, the behavior of the lgorithm may be illustrated as in Fig. (A.2) and thus is not convergent to he actual solution waveform.

Manuscript received August 18, 1986.

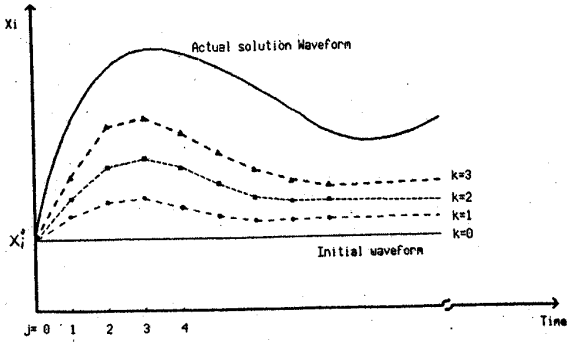


Fig. A.1. WRM iterative waveform solution behavior.

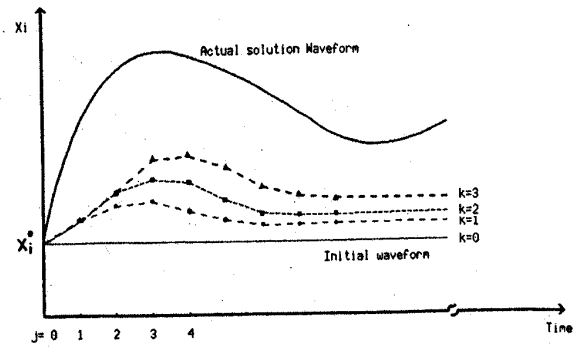


Fig. A.2. Algorithm (3.8) waveform solution behavior.

M. Ilic-Spong, M. L. Crow, and M. A. Pai: The authors thank the discussers for their valuable comments to the paper. It is clear that with the new architectures of computers, the parallel numerical algorithms will be incorporated eventually in the dynamic simulation of electric power systems. The purpose of our paper was to demonstrate the potential of such an approach. To make it practical for production grade programs much more research is required. We respond to the discussion as follows.

Razavi, Chiang, and Varaiya

i) The coherency is an inherent property of all power networks, since for a given fault certain machines tend to swing together. In other words, the machines which swing together are tightly coupled and as a group they are weakly coupled to the other groups of coherent machines. In the slow-coherency technique [9], the coherency of machines is identified independent of the fault. In [10] it is identified based on the fault dependent linearized method and demarcation of the system into study and external areas. The eigenvalues of the linearized prefault system are given as 0, 0, $\pm j3.916$, $\pm j5.953$, $\pm j6.345$, $\pm j7.034$, $\pm j7.845$, ± 8.051, $\pm j8.918$, $\pm j9.618$, and $\pm j9.722$. The number of areas is largely a matter of a clear eigenvalue separation: In this case a good separation exists between the fifth and the sixth pair of eigenvalues. The machine with the largest inertia #1, is associated with the slowest (zero) mode. Then, following the slow coherency approach [9], the other groupings are (2, 3), (4, 6, 7, 8, 10), (5), and (9). The method of [10] separates the second group further into (4, 6, 7) and (8, 10).

The latency phenomena has been observed for the rotor angles far away from the fault; however, it has not been exploited in the algorithm yet.

The question regarding mixed algebraic and differential equations is a very important one. However, due to space limitations it could not be elaborated on in detail here. A full paper on this topic is being submitted for the PICA '87 meeting.

And finally, the discussers assert that the algorithm presented by (3.8) does not converge to the correct solution. Rigorous convergence proofs are to be established yet, and moreover, since "the correct solution" is not known *a priori*, it will be premature to guess it.

However, the following is true: If equation (3.8) is derived from (3.5), one obtains

$$X_{i,k+1} = F_i(X_{1,k}, X_{2,k}, \cdots, X_{i,k+1}, \cdots, X_{n,k}) \tag{R.1}$$

or

$$\frac{X^{j+1}_{i,k+1} - X^{j}_{i,k+1}}{h} = F_i(X^{j+1}_{1,k}, X^{j+1}_{2,k}, \cdots, X^{j+1}_{i,k+1}, \cdots, X^{j+1}_{n,k}) \tag{R.2}$$

which is the Backward Euler formula for integration, generally accepted as stable numerically. Further since $X^{j+1}_{i,k+1}$ is the only unknown, we need to expand the function F_i about $X_{i,k+1}$ *only*. This implies,

$$\frac{X^{j+1}_{i,k+1} - X^{j}_{i,k+1}}{h} = F_i(X^{j+1}_{1,k}, X^{j+1}_{2,k}, \cdots, X^{j}_{i,k+1}, \cdots, X^{j+1}_{n,k})$$

$$+ \frac{\partial F_i}{\partial x_i}(X^{j+1}_{1,k}, X^{j+1}_{2,k}, \cdots, X^{j}_{i,k+1}, \cdots, X^{j+1}_{n,k})(X^{j+1}_{i,k+1} - X^{j}_{i,k-1}). \tag{R.3}$$

In this paper we have expanded F_i about *all* the variables $X_1, \cdots, X_n$ and evaluated it at the previous integration step (j) as indicated in (3.6). This also may have an effect on the true solution. It is computationally

undesirable to evaluate all of the partials. One further simplification was made (going from (3.7) to (3.8)), where the summation terms of the partial derivatives were not included. This simplification could have been the source of the assertion by the discussers as the simplification may cause the algorithm to converge to the wrong solution.

We include the results of a simulation run using these three (similar) methods:

1) Equation (3.8) of the paper:

$$X^{j+1}_{i,k+1}=X^{j}_{i,k+1}+h\left(1-h\frac{\partial F_i}{\partial x_i}(X^{j}_{1,k},\cdots,X^{j}_{i,k+1},\cdots,X^{j}_{n,k})^{-1}\right.$$
$$\cdot F_i(X^{j}_{1,k},\cdots,X^{j}_{i,k+1},\cdots,X^{j}_{n,k})$$

which needed 37.261 CPU's and at $t = 1.5$ s. The solution was $\delta_1 = 2.6773$, $\delta_2 = 2.4827$, $\delta_3 = 2.5025$, $\delta_4 = 2.5676$, and $\delta_5 = 2.3170$.

2) Equation (3.7) of the paper:

$$X^{j+1}_{i,k+1}=X^{j}_{i,k+1}+h\left(1-h\frac{\partial F_i}{\partial x_i}(X^{j}_{1,k},\cdots,X^{j}_{i,k+1},\cdots,X^{j}_{n,k})^{-1}\right.$$
$$\cdot F_i(X^{j}_{1,k},\cdots,X^{j}_{i,k+1},\cdots,X^{j}_{n,k})+\sum_{\substack{l=1\\ l\neq i}}^{n}\frac{\partial F_i}{\partial x_l}$$
$$\cdot (X^{j}_{1,k},\cdots,X^{j}_{i,k+1},\cdots,X^{j}_{n,k})(X^{j+1}_{i,k}-X^{j}_{i,k})$$

which needed 46.359 CPU's. The solution at $t = 1.5$ s was $\delta_1 = 2.8559$, $\delta_2 = 2.6611$, $\delta_3 = 2.6810$, $\delta_4 = 2.7579$, and $\delta_5 = 2.5148$.

3) Equation (R.3) of this response becomes

$$X^{j+1}_{i,k+1}=x^{j}_{i,k+1}+h\left(1-h\frac{\partial F_i}{\partial x_i}(X^{j+1}_{1,k},\cdots,X^{j}_{i,k+1},\cdots,X^{j+1}_{n,k})\right)^{-1}$$
$$\cdot F_i(X^{j+1}_{1,k},\cdots,X^{j}_{i,k+1},\cdots,X^{j+1}_{n,k})$$

which ran in 37.445 CPU's and the solution was at $t = 1.5$ s: $\delta_1 = 2.8552$, $\delta_2 = 2.6606$, $\delta_3 = 2.6804$, $\delta_4 = 2.7571$, and $\delta_5 = 2.5142$.

From these numerical results it is clear that methods 2) and 3) have a very similar solution, but method 1) is a bit different. Method 1) is the algorithm asserted not to converge to the correct solution by the discussers. However, that may be problem dependent. We wish to emphasize that the method is only one of the three methods we tested; the other two are possibly more accurate, with method 3) being superior in both accuracy and time. Thi was the method which was used for all the presented simulation result Based on the above, we believe that the thinking followed in the last part the discussion could be used as an estimate of the number of sweeps neede for the algorithm to converge with a predefined accuracy, rather than a argument of nonconvergence.

Adrián Inda

We agree that Stott's method [8] could be used to minimize the order o matrix inversion required in the INI method. We also agree that some othe partitioning techniques, such as tearing, can be applied to implemen parallel processing. In fact, the coherency-based decomposition is one o such partitioning schemes which we have used. It would be interesting t compare effects tearing techniques with the coherency-based decomposition on the efficiency of the WRM method. We plan to test it on structure preserving models.

Kundur, Rogers

The discussers have correctly inferred that the time-scale decomposition of the synchronous machine dynamics could result in the speed-up of the proposed technique. We are currently working with the detailed machine models in which at least three time scales are involved.

Further, the choice of $\Delta t = 0.00375$ s for the RK method was made to obtain the same solution as the previously published solution for this system. No grouping was applied in simulating the dynamics via the RK method.

The storage requirements of the WR method are currently high because they are programmed on a serial machine. The optimal windowing technique is important in this context to reduce storage requirements. The coherency-related questions were addressed in the reply to Razavi et al.

In summary, the authors wish to thank all the discussers for their informative comments. We believe that the power industry should exploit the potential of the parallel numerical algorithms for dynamic simulations of the system. Our work has just established the feasibility of possibly one of the most promising methods. Currently planners use transient stability programs with detailed models infrequently. Operational people run it rarely. Just as load flow through sparsity techniques has become common for both operation and planning people, it will require considerable research to exploit the new computer architectures to achieve the same goal for transient stability.

Manuscript received October 1, 1986.

Abstrac
itors prote
developed f
ances. Fau
adapted to
lations on
protection
applicable
lation has
stability p
opment and

Series
component i
sion. One
and applica
age protec
which bypa
voltage is
varistor (
the highly
zinc oxide
benefits i
transients,
greater re
prototype
were devel
last decad
protected
energized

The
protected
need for
stability
from the
which con
cycle whi
age. Thus
in the ci
protectio
on series
been fou
Because a
appeared
forward,
undertake

36 SM 357
by the I
the IEEE
at the I
Mexico,
February
May 7, 1

Printed