A multirate approach for time domain simulation of very large power systems

EINDHOVEN UNIVERSITY OF TECHNOLOGY
Department of Mathematics and Computer Science

CASA-Report 10-72
December 2010

A multirate approach for time domain simulation of very large power systems

by

V. Savcenco, B. Haut

Centre for Analysis, Scientific computing and Applications
Department of Mathematics and Computer Science
Eindhoven University of Technology
P.O. Box 513
5600 MB Eindhoven, The Netherlands
ISSN: 0926-4507

A Multirate Approach for Time Domain Simulation of very large Power Systems

Valeriu Savcenco
Technische Universiteit Eindhoven
Eindhoven, The Netherlands
V.Savcenco@tue.nl

Bertrand Haut
Tractebel Engineering S.A.
Brussels, Belgium
bertrand.haut@gdfsuez.com

Abstract - The time evolution of power systems is modeled by systems of differential and algebraic equations (DAEs) [8]. The variables involved in these DAEs may exhibit different time scales. Some of the variables can be highly active while other variables can stay constant during the entire time integration period. In standard numerical time integration methods for DAEs the most active variables impose the time step for the whole system. We present a strategy, which allows the use of different, local time steps over the variables. The partitioning of the components of the system in different classes of activity is performed automatically based on the topology of the power system. The performance of the multirate approach for two case studies is presented.

Keywords - Multirate time stepping, differential and algebraic equations, power systems simulation, numerical integration.

1 Introduction

Time evolution of power systems is modeled by large differential-algebraic systems (DAEs). These systems are built from the differential and algebraic equations describing the network, the generators, the voltage regulators, the speed governors and the dynamic loads. All together they form a non-linear system in semi-explicit form

$$\begin{aligned} y' &= f(t,y,z)\,, \\ 0 &= g(t,y,z)\,, \end{aligned} \tag{1}$$

with initial values $y(0) = y_0$ and $z(0) = z_0$, such that $g(t_0, y_0, z_0) = 0$. It is assumed that the matrix $\frac{\partial g}{\partial z}$ is non singular and therefore system (1) has index one. The joint vector of differential and the algebraic variables is denoted by $u = [y, z]^T$.

State of art numerical integration methods for DAEs use time steps that are varying in time, but are constant over the system variables [5]. Large interconnected power systems are modeled by very large DAEs of which some components may exhibit a significantly more active behavior than others, distinguishing slow and rapid temporal variations. A voltage wave propagation due to lightning lasts a few microseconds to milliseconds but a secondary frequency control may have a time duration of several minutes. A particular situation is to check the consequences of an outage. In a very large system such as the European Transmission Network most of the time the consequences of an outage are very well localized and only a few variables are impacted. Such systems can be efficiently solved using multirate methods [2, 3, 7, 9].

Multirate methods attempt to take large time steps for slowly varying components and small steps for components with a significantly more rapid variation, so as to speed up the numerical computations.

In this paper we propose a multirate time stepping approach for time integration of DAE systems describing the temporal evolution of power system networks. This approach includes the dynamic partitioning of the system variables into fast and slow and a self-adjusting strategy for the selection of the size of the time steps for all the system components.

This paper is organized as follows. In Section 2 we briefly introduce the mixed Adams-BDF method that will be used as our basic numerical integration method. In Section 3 the multirate time stepping approach is described in details. The partitioning of the variables in different classes of activity is discussed in Section 4. Results of numerical experiments for several test cases are presented in Section 5. Finally, Section 6 contains the conclusions and an outlook on further work.

2 Mixed Adams-BDF method

It is common to model the time evolution of power systems using BDF methods [4]. In this paper we will use the second-order mixed Adams-BDF method presented in [1] as our basic numerical integration method. In this method second-order Adams method is applied to the differential state variables, whereas algebraic state variables are integrated using second-order BDF method. The Adams method we use, is symmetrically A-stable (the domain of stability coincides with the left complex half-plane) and thus does not suffer from the hyper stability in contrast to the BDF method. Therefore, if the DAE system is itself unstable, the Adams method will lead to an unstable solution and will allow for detection of instabilities. We, however, still prefer to use the BDF method for the algebraic state variables, since it less sensitive to the variations in the algebraic equations than the Adams method. Detailed description and the coefficients for both methods can be found in [6].

Let us assume that we know, at time t_n, the numerical approximation of the solution u_n together with it's first two derivatives u'_n and u''_n, and we want to compute the numerical solution at time $t_{n+1} = t_n + \tau_n$. We store the vector of the solution and its derivatives in Nordsieck form

$$\boldsymbol{u}_n = \left[u_n, \tau_n u'_n, \frac{1}{2}\tau_n^2 u''_n \right]^T .$$

Following the usual predictor-corrector approach, we first compute the prediction by means of Taylor's formula

$$\begin{aligned} u_{n+1}^P &= u_n + \tau_n u_n' + \frac{1}{2}\tau_n^2 u_n'' \,, \\ u_{n+1}'^P &= u_n' + \tau_n u_n'' \,, \\ u_{n+1}''^P &= u_n'' \,. \end{aligned}$$

Next we continue with the correction stage

$$\boldsymbol{u}_{n+1} = \boldsymbol{u}_{n+1}^P + (u_{n+1} - u_{n+1}^P)\boldsymbol{l} \,,$$

where $\boldsymbol{l} = (l_0, l_1, l_2)^T$ is the vector of the coefficients of the method. Here u_{n+1} is computed from the following algebraic system

$$\begin{aligned} \tau_n y_{n+1}'^P + l_1(y_{n+1} - y_{n+1}^P) - \tau_n f(t_{n+1}, u_{n+1}) &= 0 \,, \\ g(t_{n+1}, u_{n+1}) &= 0 \,. \end{aligned}$$

For the variable step size control we need an estimate for the local error at each time step. Following [1], we estimate the local error for an attempted step from time t_n to $t_{n+1} = t_n + \tau_n$ as

$$E_{n+1} = K l_2 ||u_{n+1} - u_{n+1}^P|| \,, \qquad (2)$$

where K is a method dependent constant. In our strategy we control the L^2 norm of the error.

3 Multirate strategy

In this paper it will be assumed that the variables of the system (1) can be partitioned into fast and slow

$$y = [y_{\text{fast}}, y_{\text{slow}}]^T \quad \text{and} \quad z = [z_{\text{fast}}, z_{\text{slow}}]^T \,. \qquad (3)$$

Our multirate time stepping strategy is based on local temporal error estimation and can be described as follows. For a given global time step $\tau_n = t_{n+1} - t_n$, we first compute a tentative approximation at the time level t_{n+1} for all variables. For those components for which the error estimator indicates that smaller steps are needed, the computation is performed again with smaller time steps. At this refinement stage we perform a local variable time stepping and solve the subsystem

$$\begin{aligned} y_{\text{fast}}' &= f_{\text{fast}}(t, y_{\text{fast}}, z_{\text{fast}}, \omega) \,, \\ 0 &= g_{\text{fast}}(t, y_{\text{fast}}, z_{\text{fast}}, \omega) \end{aligned} \qquad (4)$$

where ω denotes the already computed values of the slow variables. During the refinement stage, values at the intermediate time levels of the slow components might be needed. These values can be obtained by interpolation.

The intervals $[t_n, t_{n+1}]$ are called time slabs. After each completed time slab the solutions are synchronized. In our approach, these time slabs are automatically generated, similar as in the single-rate approach, but without imposing temporal accuracy constraints on all components.

An important issue in our strategy is to determine the size of the time slabs. These could be taken large with a large multirate factor, or small with a lower multirate factor. A decision can be made based on an estimate of the number of components at which the solution needs to be calculated, including the overhead due to coupling.

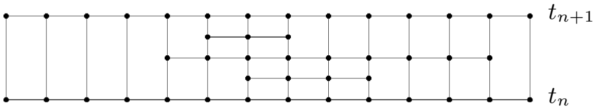

Figure 1: Example of a time slab.

In this paper we consider two levels of activity: slow variables and fast variables. One can also allow for more levels of activity. In this case, the desired accuracy does not necessary have to be achieved during the first refinement. The refinement can be continued until the error estimator is below a prescribed tolerance for all components. An example of a time slab with two levels of refinement is shown in the Figure 1.

3.1 Refinement tolerance

Let us denote the tolerance prescribed by the user by *Tol*. During the refinement stage we recompute the most active components of the system. Since the tolerance *Tol* is used to control the L^2 norm of the error for all the variables, the tolerance used during the local variable time stepping should be relaxed.

By controlling the discrete L^2 norm of the error we require

$$\left(\frac{1}{m} \sum_{i=1}^{m} e_i^2 \right)^{1/2} < \mathit{Tol} \,, \qquad (5)$$

where by e_i we denote the local error for the i-th component and m is the total number of variables. While doing the local refinements, in order to efficiently achieve the same accuracy as in the single-rate time stepping we determine

$$\mathit{Tol}_{\text{local}} = \operatorname*{argmax}_{\mathit{Tol}^*>0} \left(\frac{1}{m} \left(\sum_{i:|e_i|<\mathit{Tol}^*} e_i^2 + \sum_{i:|e_i|\geq \mathit{Tol}^*} \mathit{Tol}^{*2} \right) \right)^{1/2} \leq \mathit{Tol} \,. \qquad (6)$$

Relation (6) can also be written as

$$\mathit{Tol}_{\text{local}} = \operatorname*{argmax}_{\mathit{Tol}^*>0} \left(\sum_{slow} e_i^2 + m_{\text{fast}} \mathit{Tol}^{*2} \right) \leq m \mathit{Tol}^2 \,, \qquad (7)$$

where m_{fast} denotes the number of fast variables (the variables for which $|e_i| \geq \mathit{Tol}^*$).

In practice, there is no easy way to determine $\mathit{Tol}_{\text{local}}$ from (7). One can perform a loop from $\mathit{Tol}^* = \mathit{Tol}_{max}$ to $\mathit{Tol}^* = 1$ and stop at the first value of Tol^* which will satisfy the inequality in (7).

3.2 *Choosing the size of the time slabs*

The size of time slabs is determined automatically while advancing in time. When we are done with the processing of the n-th time slab of size Δt_n, the size of the next time slab is taken as

$$\Delta t_{n+1} = S_{n+1} \tau^*_{n+1} \,, \tag{8}$$

where S_{n+1} is the estimated multirate factor for the $(n+1)$-st time slab, and τ^*_{n+1} is the optimal time step size which would give us an estimated error smaller than the given tolerance if we were to use a single-rate approach for the next time step from t_n to $t_n + \tau^*_{n+1}$. For the first time slab we use $S_1 = 1$.

3.2.1 *Estimation of* τ^*_{n+1}

Using the information available from the n-th time slab we can estimate the value of τ^*_{n+1} for the next time slab. This is done using the standard step size control technique; the only difference is that for each component we use the information from the last available local time steps from the last time slab $[t_{n-1}, t_n]$. For example, in the time slab depicted in Figure 2, in order to estimate τ^*_{n+1}, we will use the information from the hatched areas, where the last local time steps before t_n have been taken.

Let us denote by $\hat{\tau}_n$ the size of the last time step during the local refinement. We will also assume that a numerical integration method of order p is used. In order to estimate the optimal single-rate time step τ^*_{n+1} we will use the local errors e_i for the slow variables and rescaled local errors $\left(\frac{\Delta t_n}{\hat{\tau}_n}\right)^{p+1} e_i$ for the fast ones. Here by e_i we denote the estimated local error for the variable i. The norm of this joint slow and rescaled fast errors vector will be denoted by E_n.

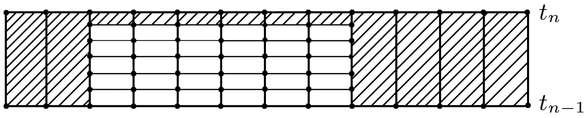

Figure 2: Time steps used for the estimation of τ^*_{n+1}.

We determine τ^*_{n+1} by

$$\tau^*_{n+1} = \vartheta \Delta t_n \sqrt[p+1]{Tol/E_n} \,, \tag{9}$$

where ϑ is a safety factor.

Expression (9) gives us an estimate of a step size with which we expect a norm of local error smaller than the tolerance.

3.2.2 *Estimation of* S_{n+1}

We assume that the multirate factor for the processed n-th time slab of size Δt_n was S_n. It means that during the local refinement in the n-th time slab we performed S_n smaller steps.

The estimation of S_{n+1} will be based on the anticipated number of fast variables. In order to estimate the optimal multirate factor we study several hypothetical computations for this last time slab. In these computations we consider what would have happened if we had taken the multirate factor larger than S_n. In particular we consider what would have happened if

$$S'_n = S_n + k \tag{10}$$

or

$$\Delta t'_n = \frac{S_n + k}{S_n} \Delta t_n \,, \tag{11}$$

for $k = 1, \ldots, k_{\text{max}}$. In our test we use $k_{\text{max}} = 10$.

The local errors can be estimates as

$$e'_i = \left(\frac{S_n + k}{S_n}\right)^{p+1} e_i . \tag{12}$$

Following the procedure from Section 3.1 for each k we determine the refinement tolerance and the number of fast components. We choose the maximum value of k for which number of fast variables is smaller than $\alpha_{\text{reject}} m$

$$S_{n+1} = S_n + \max\{0 < k < k_{\max} | m_{\text{fast}}(k) < \alpha_{\text{reject}} m\} \,. \tag{13}$$

4 Partitioning strategy

Partitioning of the variables in slow and fast can be fixed and given in advance, or it can vary in time and should be performed automatically during the time integration process. In this section we present a strategy for automatic partitioning of both differential and algebraic variables. This strategy is based on the local time variation of the numerical solution of the system and on the topology of the power system.

A power system can be usually decomposed in two parts:

- a large network which consists of a set of nodes (each node introducing two variables) connected by a set of branches (lines, cables and transformers),
- a set of components (synchronous machines, motors, loads...) which are usually connected to a particular node.

This particular structure can be used to derive a dedicated partitioning strategy.

We first perform a single step with step size τ and using an error estimator we determine the variables which do not satisfy the criterion

$$|e_i| < Tol_{\text{local}} \,, \tag{14}$$

where again e_i is the estimated local error for the variable i and Tol_{local} is the computed local tolerance. These variables will be called fast.

To allow for accurate computation of the fast variables, during the refinement stage, we also recompute the slow variables which are strongly coupled to the fast ones. The propagation of the fast status is performed as follows:

1. All the components which contain at least one fast variable are classified as fast.

2. All the nodes which contain at least one fast variable are classified as fast.

3. The connection node of a fast component is classified as fast.

4. The fast status of the nodes is then propagated through the network:

   (a) The graph G is defined as follows:

   - A node in G is defined for each electrical node;
   - An edge is defined between two nodes of G if there exists at least one branch linking the two corresponding electrical nodes;
   - A weight representing an "electrical distance" will be associated to each edge of G. Let us denote by C_1 and C_2 the two 2×2 sub-matrices of the admittance matrix coupling the pairs of variables associated nodes 1 and 2. The weight between node 1 and 2 is defined as

   $$l_{12} = \min\left(\frac{1}{\|C_1\|_\infty}, \frac{1}{\|C_2\|_\infty}\right) \qquad (15)$$

   where

   $$\|C\|_\infty = \max |C_{ij}|.$$

   (b) Each node at a distance less than a given parameter tol_G from a fast node is classified as fast.

5. All the variables belonging to a fast node or a fast component are classified as fast and will therefore be updated during the refining phase.

The creation of a table containing, for each node, the list of strongly connected nodes can be efficiently (through a modified Dijkstra algorithm and a parallel implementation) performed off-line before the start of the simulation. With this off-line preparation, the cost of the above partitioning is almost negligible during the simulation.

5 Case studies

In this section we present numerical results for two test problems. For the results reported here we used quadratic interpolation to obtain missing component values. Linear interpolation was also tried and the results were nearly identical; this simply indicates that the interpolation errors are not significant in these tests.

The computational costs are presented in terms of number of function evaluations, number of Jacobian evaluations and number of Newton iterations. We estimate the total computation cost by means of formula

$$C = 1.2 \cdot 10^{-7} N_{\text{FuncEval}} + 7.2 \cdot 10^{-7} N_{\text{JacEval}} \\ + 5 \cdot 10^{-7} N_{\text{LUFactor}} + 5 \cdot 10^{-8} N_{\text{Newton}}\,.$$

Here the coefficients represent the reference costs per variable or equation and are based on the benchmarks in a particular software package. The counters N_{FuncEval}, N_{JacEval}, N_{LUFactor} and N_{Newton} also take into account the number of the variables or the size of the matrix involved in the corespondent calculation. In our solver a LU-factorization is preformed after each update of the Jacobian, hence $N_{\text{LUFactor}} = N_{\text{JacEval}}$. From the cost coefficients it is visible that the computational cost is dominated by the cost of the Jacobian evaluations.

In our implementation of the single-rate and multirate solvers we try to reduce the number of Jacobian evaluations. A full Jacobian evaluation is performed only if it is strictly necessary:

- at the beginning of the time integration;
- the Newton method does not converge.

In all other cases (change of the time step, a discontinuity detected, etc), when necessary, we perform a local Jacobian update.

5.1 A chain test problem

For our first test problem we consider a power system composed of a chain of 100 small subsystems connected by very long lines. Each subsystem comprises a generator and the corresponding controllers modeled by 30 equations, a step-up transformer and an impedant load. A schematic illustration of the chain is presented in Figure 3. The resulting system contains 4970 variables, 3089 of which are algebraic.

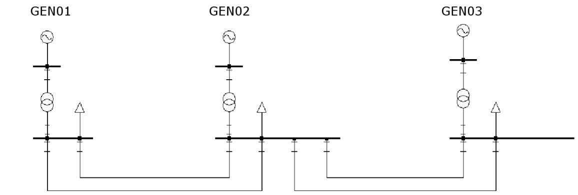

Figure 3: Chain of 100 subsystems.

A short-circuit of 100 ms is performed at the first high voltage busbar. During the very first second, this event strongly affects the beginning of the chain while the rest of the system remains more or less constant. The impact of the short-circuit propagates to the neighboring subsystems while being progressively damped.

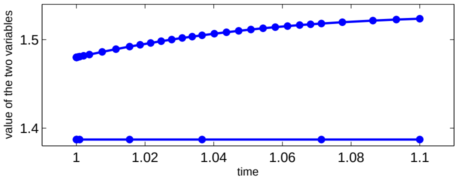

Figure 4: Solution for two components.

Figure 4 shows the time points in which the solution for two variables, one fast and one slow, were computed. It is seen that the time steps used for the fast variable are much smaller than the ones used for the slow variable. The solution of the fast variable on this interval is computed by 26 time steps, whereas only 5 time steps are needed for the slow variable. In this simulation 70 fast variables were observed.

Table 1 shows the number of function evaluations, number of Jacobian evaluations, number of Newton iterations, estimated costs and the weighted L^2- and infinity-norm errors for the single-rate and multirate methods. From these results it is seen that a substantial improvement in number of function evaluations is obtained. For the single-rate method, the number of function evaluations is four times larger. Moreover, the error behavior of the multirate scheme is very good. The speed up in terms of estimated costs is smaller than the one based on the number of function evaluations. This reduction in speed up is due to large number of Jacobian evaluations. This is again visible from the results presented in the table. An improvement of the local Jacobian evaluation within multirate time stepping is needed.

	single-rate	multirate
$\|\|\text{error}\|\|_2$	$4.22 \cdot 10^{-5}$	$4.22 \cdot 10^{-5}$
N_{FuncEval}	184326	47102
N_{JacEval}	11892	15355
N_{Newton}	184326	47102
C	0.045	0.026

Table 1: Errors and computational costs for the chain problem.

5.2 *PEGASE problem*

As the second test we consider the PEGASE problem. This problem is a dedicated test case constructed by the PEGASE consortium [10]. The system modeled is loosely inspired from the European transmission grid in terms of size (number of branches, nodes, generators, loads), topology and type of units (nuclear, hydro, TGV). The problem is modeled by a DAE system with 123465 variables, of which 50235 are algebraic.

The main features of the PEGASE test system are

- 15350 buses.
- 3824 synchronous machines with generic models of AVRs, speed governors and turbines.
- 4853 dynamic loads. Some of them include an equivalent of the distribution transformer and medium-voltage feeder.

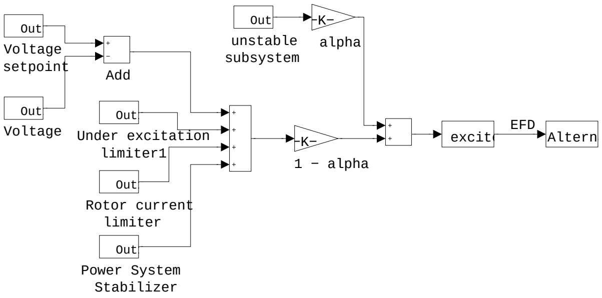

Figure 5: Diagram of the disturbtion of a machine's excitation controller.

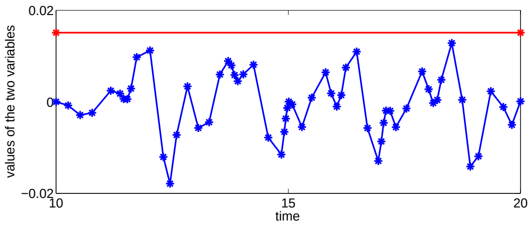

Figure 6: Time evolution of one fast and one slow variable.

	single-rate	multirate
$\|\|\text{error}\|\|_2$	$2.07 \cdot 10^{-3}$	$6.87 \cdot 10^{-4}$
N_{FuncEval}	29014275	798470
N_{JacEval}	2800147	274167
N_{Newton}	30866250	555454
C	8.44	0.45

Table 2: Errors and computational costs for the PEGASE problem, $\alpha = 0.2$.

	single-rate	multirate
$\|\|\text{error}\|\|_2$	$4.88 \cdot 10^{-4}$	$4.80 \cdot 10^{-4}$
N_{FuncEval}	7284435	618885
N_{JacEval}	263660	263764
N_{Newton}	7037505	372033
C	1.54	0.47

Table 3: Errors and computational costs for the PEGASE problem, $\alpha = 0$.

	single-rate	multirate
$\|\|\text{error}\|\|_2$	$1.55 \cdot 10^{-3}$	$4.29 \cdot 10^{-4}$
N_{FuncEval}	30125460	13199491
N_{JacEval}	3087649	3382476
N_{Newton}	32100900	13006130
C	8.98	6.35

Table 4: Errors and computational costs for the PEGASE problem, $\alpha = 0.5$.

We solve this problem on the time interval $0 < t < T = 20$. One of the machines has its excitation controller disturbed by a local sustained instability (see Figure 5). The parameter α characterizes the extent of this disturbance with respect to the classical control loop. If $\alpha = 0$ then the machine is correctly controlled and the instability should not affect the network. If $\alpha = 1$ then the machine is steered in open-loop by the local instability which will strongly affect the network. In our test we consider $\alpha = 0.2$. We expect that this event will only have a local impact and hence, multirate method will be able to exploit this difference in the time scales.

Figure 6 shows the time points in which the solution for two variables, one fast and one slow, were computed during the time interval when the disturbance occurred. It is seen that the time steps used for the fast variable are much smaller than the ones used for the slow variable. The solution for the fast variable on this interval is computed by 46 time steps, whereas only 1 time step is needed for the slow variable.

Table 2 shows the number of function evaluations, number of Jacobian evaluations, number of Newton iterations, estimated cost (in seconds) and the weighted L^2-norm error (measured with respect to an accurate reference solution) for the global time interval $[0, T]$ for the single-rate and multirate methods. From these results it is seen that a substantial improvement in cost is obtained. For the single-rate method the estimated costs are twenty times larger. Moreover, the error behavior of the multirate scheme is very good.

For comparison, in Table 3 and Table 4, results for two additional values of the parameter α are presented. For the test case with $\alpha = 0$ the disturbance is isolated and does not propagate through the network. For this case the number of function evaluations and number of Newton iterations required by the single-rate solver are considerably larger than for the multirate solver. Due to the use of local Jacobian updates, the total number of Jacobian evaluations is similar for multirate and single-rate solver for this value of α. This is also the reason why the speed-up in the total computation time is not so large as for $\alpha = 0.2$. For the test case with $\alpha = 0.5$ the disturbance strongly affects the network making most of the variables active. No significant speed up is achieved in this case.

6 Conclusions

In this paper we presented a multirate time stepping strategy for systems of differential and algebraic equations resulting from modeling of power systems. The algorithm for dynamic partitioning of the components into slow and fast was described. Numerical experiments confirmed that the efficiency of time integration methods can be significantly improved by using large time steps for inactive components, without sacrificing accuracy.

Acknowledgment

This work was performed in the context of the PEGASE project[10] funded by European Community's 7th Framework Programme (grant agreement No. 211407).

REFERENCES

[1] J.Y. Astic, A. Bihain and M. Jerosolimski, "The mixed Adams-BDF variable step size algorithm to simulate transient and long term phenomena in power systems", IEEE Trans. Power Systems, 9, 929–935, 1994.

[2] A. Bartel and M. Günther, "A multirate W-method for electrical networks in state space formulation", J. Comp. Appl. Math. 147, 411–425, 2002.

[3] J. Chen and M.L. Crow, "A variable partitioning strategy for the multirate method in power systems", IEEE Trans. Power Systems, 23, 259–266, 2008.

[4] C.W. Gear, "Numerical initial value problems in ordinary differential equations", Englewood Cliffs, N.J.Prentice Hall, 1971.

[5] E. Hairer, S.P. Nørsett and G. Wanner, "Solving Ordinary Differential Equations I – Nonstiff Problems", Second edition, Springer Series Comput. Math. 8, Springer, 1993.

[6] E. Hairer and G. Wanner, "Solving Ordinary Differential Equations II – Stiff and Differential-Algebraic Problems", Second edition, Springer Series in Comp. Math. 14, Springer, 1996.

[7] V. Savcenco, W. Hundsdorfer and J.G. Verwer, "A multirate time stepping strategy for stiff ODEs", BIT, 47, 137–155, 2007.

[8] M. Stubbe, A. Bihain, J. Deuse and J.C. Baader, "Simulation of the dynamic behaviour of electrical power systems in the short and long terms.", CIGRE, 38-03, 1998.

[9] A. Verhoeven, B. Tasic, T. Beelen, E.J.W. ter Maten and R.M.M. Mattheij. "Automatic partitioning for multirate methods", In G. Ciuprina, D. Ioan (Eds), Scientific Computing in Electrical Engineering, Springer, Berlin Heidelberg New York, 229–236, 2007.

[10] [Online], Available at http://www.fp7-pegase.eu/.

PREVIOUS PUBLICATIONS IN THIS SERIES:

Number	Author(s)	Title	Month
10-68	J.H.M. Evers A. Muntean	Modeling micro-macro pedestrian counterflow in heterogeneous domains	Nov. '10
10-69	S.W. Rienstra M. Darau	Boundary layer thickness effects of the hydrodynamic instability along an impedance wall	Nov. '10
10-70	T. Fatima A. Muntean	Sulfate attack in sewer pipes: Derivation of a concrete corrosion model via two-scale convergence	Nov. '10
10-71	M.E. Rudnaya R.M.M. Mattheij J.M.L. Maubach	Orientation identification of the power spectrum	Dec. '10
10-72	V. Savcenco B. Haut	A multirate approach for time domain simulation of very large power systems	Dec. '10

Ontwerp: de Tantes,
Tobias Baanders, CWI