Variable Step Size Time Integration Methods for Transient Eddy Current Problems

Frank Cameron, Robert Piché and Kimmo Forsman
Tampere University of Technology, P.O. Box 692, 33101 Tampere, Finland

***Abstract*—For transient eddy current problems modelled as differential-algebraic equations (DAEs) a time integration method suitable for ordinary differential equations (ODEs) will not necessarily work. We present two Runge-Kutta methods that are suitable for the time integration of the classes of DAEs to which eddy current problems belong. Both methods have error estimators and hence allow variable step sizes. In tests our variable step size integrators were competitive with fixed step size integrators, in particular with Crank-Nicolson.**

***Index terms*— Eddy currents, Electromagnetic transient analysis, Integrodifferential equations, Runge-Kutta methods**

I. Introduction

Common numerical approaches for solving nonlinear transient eddy current (EC) problems yield, after spatial discretization, a system of N equations of the form

$$\mathbf{S}(\cdot)\,\mathbf{y}' \;=\; \mathbf{R}(\mathbf{y},t)\,\mathbf{y} + \mathbf{f}(t)\;,\quad \mathbf{y}(t_0) = \mathbf{y}_0 \qquad (1)$$

In (1) $\mathbf{y}(t)$ is the vector of states (i.e. degrees of freedom), $\mathbf{y}' = d\mathbf{y}/dt$ and $\mathbf{f}(t)$ is a driving term. The choice of an appropriate time integration method for (1) depends on the properties of $\mathbf{S}$ (see Table I). EC problems are challenging because $\mathbf{S}$ is singular and thus (1) is not an ODE but rather a system of differential-algebraic equations (DAEs).

Depending on how it is formulated, an EC problem may belong to class C_2 or to class C_3. If the objects are stationary and the only source of nonlinearity is caused by the permeability $\mu = \mu(|H|)$, H-oriented formulations (implying the tangential continuity of the magnetic field H) belong to class C_3 and B-oriented formulations (implying the normal continuity of the magnetic flux density B) to class C_2. If the conductivity σ also is nonlinear then practically all formulations belongs to class C_3, see e.g. [1]–[3]. Problems with moving objects will in general belong to C_3, i.e. $\mathbf{S}$ will depend on t.

Time integration methods designed for ODEs may not be appropriate for DAEs, in particular they may lose *order*. A time integration method is of order p if the local error depends asymptotically on the time step size h as $O(h^{p+1})$. Let $p(C_i)$ be the order of a method for C_i. Then it is known that $p(C_1) \geq p(C_2) \geq p(C_3)$. With regards to order, a method designed for C_3 should also be suitable for C_2 and C_1.

Manuscript received November 3, 1997.
Frank Cameron, e-mail frank@cc.tut.fi

An important issue in time integration is whether the step size h is constant or variable. With variable h one has the advantage of being able to control the error incurred in computing the time integration. Variable h schemes however are more difficult to design and implement.

Nicolet and Delincé [4] raised the issues of the DAE nature of EC problems and variable vs. constant h. The time integration methods they proposed were only suitable for class C_2 problems. We wish to solve class C_3 problems. In addition the variable h scheme proposed by Nicolet and Delincé has the lowest possible order. Our purpose is to design and implement variable step size time integration methods for class C_3 whose order is higher than that of Nicolet and Delincé.

II. Runge-Kutta methods

We use time integration methods from the Runge-Kutta (RK) family. An s stage RK method is specified by an $s \times s$ matrix $\mathbf{A}$ and $s \times 1$ vector $\mathbf{b}$. Starting from the initial condition $\mathbf{y}_0$ at t_0 the ith stage of an RK method is defined by

$$\mathbf{Y}_i = \mathbf{y}_0 + h\sum_{j=1}^{s} a_{ij}\mathbf{Y}'_j\;. \qquad (2)$$

The RK method is applied by substituting $\mathbf{Y}_i$ for $\mathbf{y}$ and $\mathbf{Y}'_i$ for $\mathbf{y}'$ in (1) and solving the resulting equation set. When all stages have been computed the state is updated using

$$\mathbf{y}_1 = \mathbf{y}_0 + h\sum_{i=1}^{s} b_i\mathbf{Y}'_i\;. \qquad (3)$$

To find $\mathbf{Y}_i$, $i = 1, 2, \ldots, s$ when $\mathbf{A}$ is full we must solve a set of $N \cdot s$ equations. There are several ways of avoiding having to solve this large set: (a) restrict $\mathbf{A}$ to be lower triangular, (b) use a triangular approximation to $\mathbf{A}$ [5], or (c) factorize $\mathbf{A}$ using a Jordan or Schur decomposition and rewrite the equations to be solved [6, sec IV.8]. We have chosen option (a) since it is the easiest to implement.

TABLE I
Classes of eq. (1).

Class	Properties of $\mathbf{S}(\cdot)$	Type
C_1	constant, nonsingular	ODE
C_2	constant, singular	DAE
C_3	depends on $\mathbf{y}$, t, singular	DAE

An *embedded pair* of RK methods is typically used to estimate the error in $\mathbf{y}$. Such a pair uses the same $\mathbf{A}$ matrix but different advancing vectors $\mathbf{b}$ and $\widehat{\mathbf{b}}$. Assuming $\widehat{\mathbf{b}}$ yields the more accurate RK method, i.e. the method with higher order, then an estimate of the *local* error in $\mathbf{y}$ from (3) is

$$\mathbf{e} = h \sum_{i=1}^{s} (b_i - \hat{b}_i) \mathbf{Y}'_i \ . \tag{4}$$

The local error is the error produced in one time step by a time integration method assuming exact initial conditions. The global error is the difference at any time between the exact value of $\mathbf{y}$ and the estimate of $\mathbf{y}$ provided by a time integration method. Thus global error takes into account the accumulation of error as we move step by step.

The errors discussed here are solely those due to time integration, i.e. we do not consider errors due to the spatial discretization.

We have designed two embedded pairs using the theory of Kvaernø [7] and Jay [8]. For the first pair

$$\mathbf{A} = \frac{1}{6} \begin{bmatrix} 6-3\sqrt{2} & 0 & 0 & 0 \\ 3\sqrt{2} & 6-3\sqrt{2} & 0 & 0 \\ 30-18\sqrt{2} & 24\sqrt{2}-36 & 6-3\sqrt{2} & 0 \\ 2\sqrt{2}+1 & -2+\sqrt{2} & 1 & 6-3\sqrt{2} \end{bmatrix} \tag{5}$$

The $\mathbf{b}$ and $\widehat{\mathbf{b}}$ vectors are equal to the 2nd and 4th rows of $\mathbf{A}$ respectively. For the second pair

$$\mathbf{A} = \frac{1}{2088000} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 801792 & 801792 & 0 & 0 \\ 970833 & 315375 & 801792 & 0 \\ 590875 & 1953125 & -1257792 & 801792 \end{bmatrix} \tag{6}$$

The $\mathbf{b}$ and $\widehat{\mathbf{b}}$ vectors are equal to the 3rd and 4th rows of $\mathbf{A}$ respectively. For both (5) and (6) it can be shown that the $\mathbf{b}$ and $\widehat{\mathbf{b}}$ methods have orders of $O(h^2)$ and $O(h^3)$ respectively for class C_3.

Method (5) has 4 implicit stages, i.e. 4 nonzero diagonal elements, while (6) has 3 implicit stages. Depending on the EC problem in question, this difference may be significant since each implicit stage implies a set of (non)linear equations to be solved. However, methods where the first row of $\mathbf{A}$ is zero have the disadvantage that for class C_3 they are not "self-starting": either the first step must be done using another RK method or $\mathbf{y}'$ at t_0 must be given.

Although (4) only estimates the error of the lower order method, we are free to update $\mathbf{y}$ with either the lower or higher order method. We update $\mathbf{y}$ with the higher order method, i.e. using $\widehat{\mathbf{b}}$. We shall call the RK method based on (5) **Cam1** and that based on (6) **Cam2**.

Both the $\mathbf{b}$ and $\widehat{\mathbf{b}}$ methods of (5) are L-stable. The $\mathbf{b}$ and $\widehat{\mathbf{b}}$ methods of (6) are A-stable and have asymptotic stability function values of roughly 0.81 and 0.41 respectively. Hairer and Wanner's book [6, sec IV.3] describes these stability measures.

With variable step size methods we cannot choose the time instants at which $\mathbf{y}$ is computed. So it is useful to have interpolators to compute $\mathbf{y}$ at any desired time instants. With RK methods we can use the stage values $\mathbf{Y}_i$, $i = 1, 2, \ldots, s$ to design interpolators that are computationally cheap. We have designed RK interpolators of order $O(h^2)$ for both (5) and (6). Aside from their use in computing output at desired time instants the interpolators can be used to predict starting values for the non-linear equations.

III. Crank-Nicolson, Midpoint and Trapezoid

There are actually several methods in the literature, differing slightly from one another, that are called "Crank-Nicolson". We discuss two RK methods that can be interpreted as Crank-Nicolson methods: (a) the *implicit midpoint* rule and (b) the *trapezoid* rule. To see the difference between these two methods consider (1) without a driving term and with constant $\mathbf{S}$. For the first step the implicit midpoint rule yields

$$\mathbf{S} \cdot (\mathbf{y}_1 - \mathbf{y}_0) \ = \ \frac{h}{2} \mathbf{R} \left(\frac{1}{2} (\mathbf{y}_1 + \mathbf{y}_0) \right) \cdot (\mathbf{y}_1 + \mathbf{y}_0) \tag{7}$$

while the trapezoid rule yields

$$\mathbf{S} \cdot (\mathbf{y}_1 - \mathbf{y}_0) \ = \ \frac{h}{2} \Big(\mathbf{R}(\mathbf{y}_1) \mathbf{y}_1 + \mathbf{R}(\mathbf{y}_0) \mathbf{y}_0 \Big) \tag{8}$$

The nonlinear $\mathbf{R}$ is clearly handled differently by the two methods. If $\mathbf{S}$ depended on $\mathbf{y}$ it would also be handled differently. While both methods have order $O(h^2)$ for class C_1, only the trapezoid rule is $O(h^2)$ for classes C_2 and C_3. As our EC problems belong to either C_2 or C_3 we have not used the midpoint rule.

IV. Error control

The step size h is adjusted to maintain an error measure ϵ below a certain level. The error measure we have used is a root-mean-square expression,

$$\epsilon = \left(\frac{1}{N} \sum_{i=1}^{N} \Big(\frac{|e_i|}{r\,|y_i| + \nu_i} \Big)^2 \right)^{1/2} , \tag{9}$$

where e_i is the local error estimate from (4), r is a user-requested relative tolerance and ν_i is a "threshold" value for component y_i. We adjust h to maintain $\epsilon \leq 1$.

Threshold ν_i represents a value that is "small" but nonzero for y_i. If ν_i were absent, problems would occur in (9) when $y_i \approx 0$. Thus far we have used previous runs to get estimates for ν_i. We are working on a procedure that automatically estimates ν_i using a few RK steps before the actual run starts.

The basic rule for the controller that adjusts h is as follows: if $\epsilon < 1$ then increase h else if $\epsilon > 1$ then reduce h. We have implemented Gustafsson's PI-controller [9] for adjusting h.

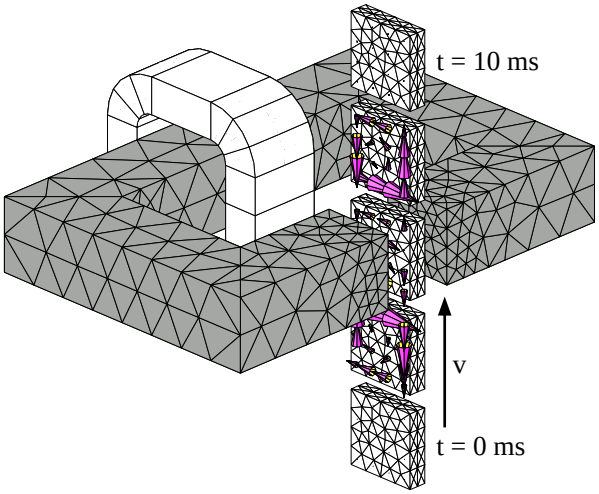

Fig. 1. The test system.

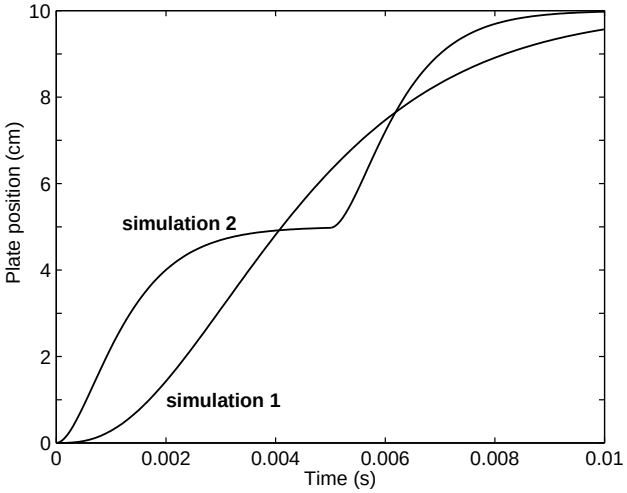

Fig. 2. The plate position for simulations 1 and 2.

V. Tests

Our test system (Fig. 1) consisted of a driving coil, a non-conducting iron yoke and a conducting, non-magnetic plate. The current in the coil is constant and we assume constant permeability for iron, $\mu_r = 1000$, and constant conductivity $\sigma = 2.0 \cdot 10^7$ S/m for the conducting plate.

We imposed movement on the plate: it starts below the yoke from rest, moves through the gap between the poles and finishes above the yoke. Figure 1 shows the plate at five different positions along its path of movement. As intuition would suggest the eddy currents in the plate reverse their direction of circulation as the plate moves through the gap.

For this test system we used the H-oriented hybrid formulation presented in [10] and [3]. In this formulation only the magnetic and conducting regions are modelled (i.e. included in the domain Ω) and thus the movement of the plate is easily realized. Whitney edge elements are used for approximating the magnetic field H and thus a state y_i is the circulation of H along edge i of the mesh. The test system is a linear time-varying problem since μ and σ are constant. After taking symmetry into account our spatial discretization has 451 nodes and 1549 tetrahedra and $\mathbf{y}$ has 597 states.

We conducted two simulations that differed in the motion imposed on the plate (Fig. 2). In both simulations we estimated the response in four different ways:

Case 1 fixed h, Trapezoid rule
Case 2 fixed h, $\mathbf{b}$ method from (5)
Case 3 variable h, Cam1 from (5)
Case 4 variable h, Cam2 from (6)

In Case 2 since $\mathbf{b}$ only needs the first two stages we did not use the entire $\mathbf{A}$ of (5), but only first two rows. We computed global accuracy based on the total ohmic power losses in the conducting plate:

$$P = \int_{\Omega_c} \frac{1}{\sigma} \text{curl}\, H \cdot \text{curl}\, H = \mathbf{y}^T \mathbf{D} \mathbf{y}, \qquad (10)$$

where the domain Ω_c includes all conducting parts of the system. The matrix $\mathbf{D}$ is defined by

$$d_{ij} = \int_{\Omega_c} \frac{1}{\sigma} \text{curl}\, w_i \cdot \text{curl}\, w_j,$$

where w_i is the edge element basis function associated with edge i. The accuracy we used is

$$\text{Accuracy} = P_{max} \cdot \left(\max_{t_i} | \, P_{exact}(t_i) - P_{est}(t_i) \, | \right)^{-1} \qquad (11)$$

In (11) P_{max} is the largest power observed, P_{est} is an estimate computed from one of the four Cases and P_{exact} is obtained using a very small step size h.

Figure 3 shows the power response for the first simulation. The dotted vertical line shows the time when the plate is directly in the gap between the poles. The work-accuracy diagram for the four different cases is Fig. 4. Figures 5 and 6 are the corresponding figures for the second simulation. In the second simulation the plate almost stops when it is directly in the gap between the poles.

As Nicolet and Delincé [4] point out, it is difficult to compare fixed and variable step size methods. The re-

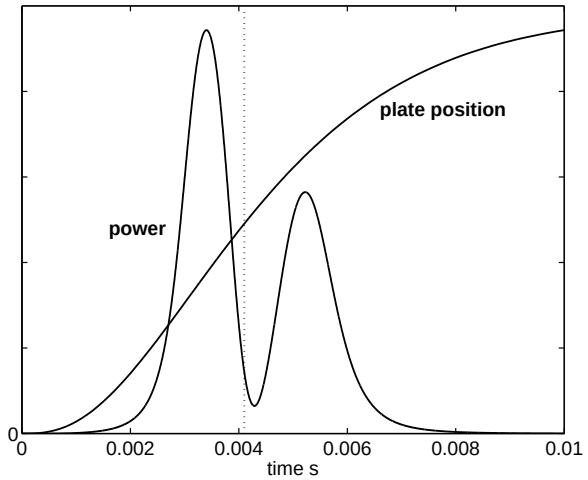

Fig. 3. The power response for simulation 1. Solid line: exact solution, +: points computed by Case 4, dashed line: Case 4 RK interpolator.

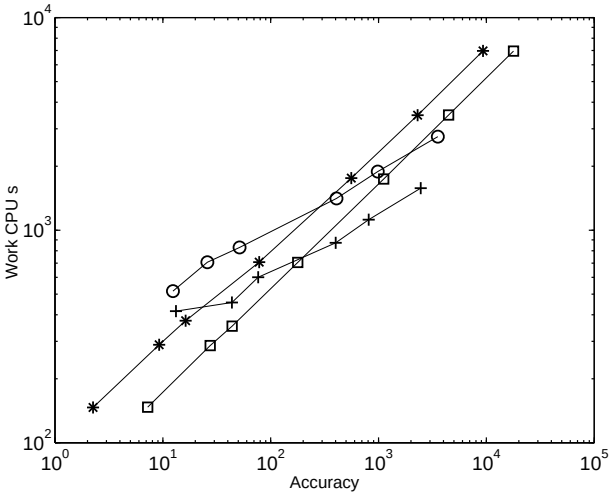

Fig. 4. Work vs. accuracy for simulation 1. □: Case 1, *: Case 2, ○: Case 3, +: Case 4.

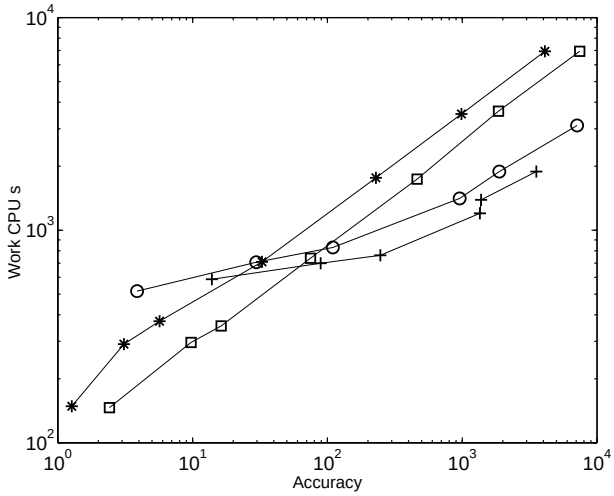

Fig. 6. Work vs. accuracy for simulation 2. □: Case 1, *: Case 2, ○: Case 3, +: Case 4.

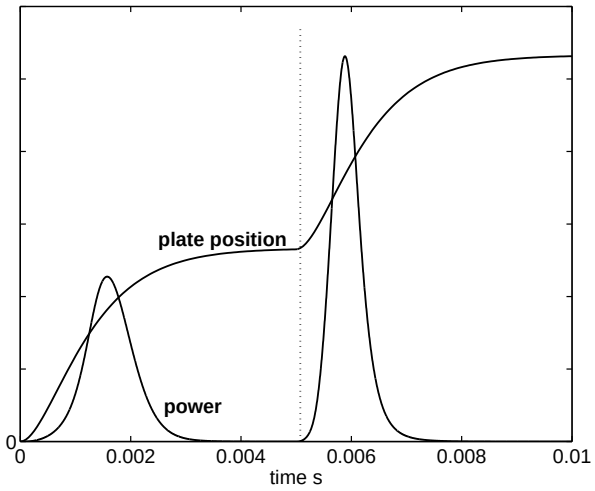

Fig. 5. The power response for simulation 2. Solid line: exact solution, ○: points computed by Case 3, dashed line: Case 3 RK interpolator.

sults one gets are very much dependent on the simulation parameters: the input, the length of the simulation, what error criteria are used and so on. Nonetheless we can make some observations based on the above results:

- The variable step methods are better than the fixed step methods when higher accuracies are demanded.
- Cam2 (6) performs better than Cam1 (5).
- The variable step methods are better than the fixed step methods in simulations where there are periods of "fast" and "slow" dynamics.
- The RK interpolators work well in providing points between the times when the RK methods compute the states $\mathbf{y}$.

The final observation is based on the power responses produced by interpolators corresponding to Cam1 and Cam2 shown in Figures 3 and 5. These responses lie almost on top of the exact power response even though they correspond to runs with relatively low accuracy.

VI. Conclusion

We have developed two variable step size Runge-Kutta methods for the classes of DAEs to which eddy current problems belong. In addition both RK methods have corresponding RK interpolators that can cheaply compute state values at any desired time. Comparing variable step size integrators and fixed step size integrators in terms of CPU time and accuracy is difficult; the quantitative results one gets will depend on the nature of the simulation, on the accuracy measures used and so on. However in qualitative terms a variable step size integrator has something a fixed size integrator can never have: a facility for controlling error due to time discretization. We should be willing to pay some CPU time for this facility.

References

[1] O. Biró, K. Preis, and K. R. Richter, "Various fem formulations for the calculation of transient 3d eddy currents in nonlinear media," *IEEE Trans. Magn*, vol. 31, pp. 1307–1312, May 1995.

[2] R. Albanese, F. I. Hantila, and G. Rubinacci, "A nonlinear eddy current integral formulation in terms of a two-component current density vector potential," *IEEE Trans. Magn.*, vol. 32, no. 3, pp. 784–787, 1996.

[3] T. Tarhasaari, A. Koski, K. Forsman, and L. Kettunen, "Hybrid formulations for the eddy current problem with moving objects," *IEEE Trans. Magn.*, in press.

[4] A. Nicolet and F. Delincé, "Implicit Runge-Kutta methods for transient magnetic field computation," *IEEE Trans. Magn*, vol. 32, pp. 1405–1408, May 1996.

[5] P. J. van der Houwen and J. J. B. de Swart, "Triangularly implicit iteration methods for ODE-IVP solvers," *SIAM J. Sci. Statist. Comput.*, vol. 18, pp. 41–55, 1997.

[6] E. Hairer and G. Wanner, *Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems.* Berlin: Springer-Verlag, 1991.

[7] A. Kvaernø, "Runge-Kutta methods applied to fully implicit differential-algebraic equations of index 1," *Math. Comput.*, vol. 54, pp. 583–625, 1990.

[8] L. Jay, "Convergence of a class of Runge-Kutta methods for differential-algebraic systems of index 2," *BIT*, vol. 33, pp. 137–50, 1993.

[9] K. Gustafsson, *Control of Error and Convergence in ODE Solvers.* Ph.D. thesis, Lund Institute of Technology, 1992.

[10] K. Forsman and L. Kettunen, "Hybrid formulations for 3D magnetostatic and eddy current problems," *ACES Journal*, vol. 12, pp. 140–144, July 1997.