Explicit Exponential Method for the Integration of Stiff Ordinary Differential Equations

Sami S. Ashour* and Owen T. Hanna†
University of California, Santa Barbara, Santa Barbara, California 93106

A new procedure for the integration of both stiff and nonstiff ordinary differential equations is presented. This new approach, which is applicable to general nonlinear systems, is fully explicit, requires little computer memory, and is very easy to program. It does not require computation of a Jacobian matrix or the solution of algebraic equations. The new algorithm has much better stability properties than the classical explicit methods and is generally much more efficient than these methods for stiff problems. The new approach automatically partitions the dependent variables at every step into stiff and nonstiff groups, and subsequently integrates each accordingly, using a nonstiff method for the nonstiff variables and a new explicit exponential method for the stiff variables. The algorithm blends these approximations appropriately. Because of the automatic partitioning, the new algorithm is entirely suitable for both stiff and nonstiff problems.

Nomenclature

$d\text{old}_i$ = derivative value for ith variable at start of RK2 step
$d\text{new}_i$ = derivative value for ith variable at end of RK2 step
$F(s)$ = arbitrary function in Eq. (2)
$f(t,y)$ = derivative vector of $y, y' = f(t,y)$, $[f_1(t,y_1, \ldots, y_N), \ldots, f_N(t,y_1, \ldots, y_N)]^T$
h = integration step size
h_0 = initial integration step size
m_i = exponential or "stiffness" parameter for ith variable
N = number of first-order ordinary differential equations in system
t = independent variable
$w_i(t)$ = temporary local exponentially fitted solution for ith variable, used to estimate local error for the exponential method, Eq. (7)
$y(t)$ = solution vector for the dependent variables, $(y_1, \ldots, y_N)^T$
y_0 = vector of initial values for the dependent variables at initial point t_0, $[y_1(t_0), \ldots, y_N(t_0)]^T$
$z(t)$ = temporary local Euler solution vector, used to estimate local error for RK2
λ = positive constant in prototype equation, $y' = -\lambda y$

Introduction

THE classical explicit integration methods (such as explicit Runge-Kutta and Adams methods, etc.) are known to have very limited stability regions.[1] Thus, whenever the integration step size is restricted mainly by stability rather than by accuracy, the user is forced to take very small step sizes when using such methods, and they are extremely inefficient.[1] When the step size is controlled mainly by accuracy, the problem is said to be nonstiff; conventional explicit methods are the most efficient methods for nonstiff problems. Problems for which the step size is controlled by stability considerations are termed stiff. Practitioners usually find it necessary to use implicit methods (such as implicit Runge-Kutta and backward-differentiation methods) to maintain acceptable stability when integrating stiff problems.[1] But implicit methods require the solution of nonlinear systems of algebraic equations. To accomplish this, Newton's method (or a variant of it) is usually used.[1] This requires the Jacobian matrix, which occupies a large space in the computer memory and is expensive to evaluate (especially for very large problems). Hence the cost per step for implicit methods is much higher than that of explicit methods. Even more importantly, the implicit methods require much more storage in the computer memory (especially for very large problems) than the explicit methods. Thus, for those real-time applications where the computer is in charge of performing various tasks continuously and there is very limited memory available for the integration code, the implicit codes may not be suitable. For such situations, it is desirable to have an explicit method that is more efficient than the classical explicit methods. It would also be very convenient to have a method that could automatically determine whether the problem at hand is stiff or nonstiff and be able to integrate both types efficiently. Such a method will be developed in this paper.

Basic Method

The discussion is addressed to the customary exponentially stiff problem. The reason some implicit methods have very good stability properties is that they effectively provide a kind of rational approximation to the solution, and such an approximation can accommodate both the polynomial and the rapidly decaying exponential behavior that are encountered simultaneously in a stiff problem. On the other hand, the computational disaster experienced by classical explicit methods in stiff problems is associated with the inappropriate (growing) polynomial approximation to a rapidly decaying exponential. In this situation, a rapidly decaying component having a small magnitude is approximated by a function with a growing magnitude, leading to a catastrophic relative error and the contamination of other solution components.

The method proposed here to solve this dilemma is to identify any locally exponentially decaying variables (stiff variables) at the beginning of every step of the computation, and to approximate them in an appropriate way that blends properly with approximations for the nonexponentially decaying variables (nonstiff variables). To do this, we formulate the differential equation solution as a simple integral equation for each component. The new method then achieves the above objective through a combination of 1) classical series approximations for the polynomial components, and 2) doubly asymptotic (and exponentially improved) approximation by Watson's Lemma[2] for the exponential components. These two integration procedures are then properly combined into a sim-

Received Jan. 14, 1990; revision received Aug. 3, 1990; accepted for publication Aug. 9, 1990.

*Graduate Research Assistant.
†Professor, Department of Chemical and Nuclear Engineering.

ple and effective general algorithm. This algorithm has a straightforward theoretical justification and appears to work well in various examples where explicit methods have failed.

To develop the method, we consider a single ordinary differential equation (ODE) of the form $y' = f(t,y)$; extension to systems follows directly. This equation is now formally integrated from t to $(t + h)$, resulting in the following integral equation for $y(t)$:

$$y(t + h) = y(t) + \int_{s=t}^{t+h} f[s,y(s)]\, ds \tag{1}$$

Equation (1), which is the fundamental theorem of integral calculus, can be taken as the starting point for various results in the theory of approximation. We now illustrate a very useful way to employ Eq. (1) when y is in a region of strong exponential decay (where classical explicit integration methods encounter stability difficulties). If we locally linearize $y' = f(t,y)$ near (t_0,y_0) and if $(\partial f/\partial y)(t_0,y_0)$ is negative and large (in magnitude), then we have strong exponential decay. The local solution to the differential equation is then just the homogeneous decaying exponential plus a complementary term that is merely a linear function of t. Thus, the derivative $y'(t)$ of this local solution has the same strong exponential decay (plus a constant term). The essential point is that when the solution is behaving stiffly, the derivative can be expressed as a strongly decaying exponential times a function that is locally varying slowly. This fact, when used with a doubly asymptotic approximation of the integral in Eq. (1), furnishes a useful stable integration of any locally stiff variables. It should be noted that, although the above discussion of locally exponential behavior of the solution derivative is based on a single equation, entirely analogous results apply for systems of coupled nonlinear ordinary differential equations. The computer algorithm and the results reported later in this paper are for such systems.

To implement the above ideas properly, we need to introduce the concept of exponential improvement of asymptotic expansions of Laplace integrals of the type:

$$I = \int_0^A e^{-xs} F(s)\, ds, \qquad x > 0 \tag{2}$$

where $F(s)$ is of exponential order. For a fixed upper limit A, where x is large, the integral has an asymptotic expansion in x obtained by expanding $F(s)$ in a Maclaurin series and integrating term by term from 0 to ∞ (ignoring the upper limit A). This result is known as Watson's Lemma.[2] When the upper limit A is too small relative to x for this procedure to be useful, we can obtain an expansion that is asymptotic, either for $x \to \infty$ with A fixed, or for $A \to 0$ with x fixed, by merely retaining the upper limit in the integration when $F(s)$ is expanded.[3] This technique plays a central role in what follows.

An exponential improvement to the asymptotic expansions of Eq. (2) (useful for either $x \to \infty$ or $A \to 0$) is accomplished as follows. We write

$$F(s) = e^{-ms} F(s) \cdot e^{ms} = e^{-ms}\{F(0) + [F'(0) + mF(0)]s + \mathcal{O}(s^2)\}$$

If we take $[F'(0) + mF(0)] = 0$, then the $\mathcal{O}(s)$ term in the Maclaurin expansion is zero and an effective two-term expansion of I in Eq. (2) (for either $x \to \infty$ or $A \to 0$) is

$$I = \int_0^A e^{-(x+m)s} F(s) \cdot e^{ms}\, ds \approx \frac{F(0)}{(x+m)} (1 - e^{-(x+m)A}) \tag{3}$$

It is easy to show that Eq. (3) reduces to the appropriate two-term approximations to Eq. (2) for either $x \to \infty$ or $A \to 0$. A version of Eq. (3) will now be used to accomplish the exponential integration of any apparent local stiff variation of y in Eq. (1).

To proceed, then, we return to Eq. (1) and write it as (with the change of variables $s = t + u$)

$$y(t + h) = y(t) + \int_{u=0}^{h} e^{-mu} y'(t + h) e^{mu}\, du \tag{4}$$

The results of the preceding paragraph are valid only if m is positive. To determine m, we first expand $[y'(t + u)e^{-mu}]$ as $\{y'(t) + [y''(t) + my'(t)]u + \mathcal{O}(u^2)\}$. We choose m so that $[y''(t) + my'(t)] = 0$, which gives $m = -y''(t)/y'(t)$. This choice is crucial because it amounts to locally fitting the decaying transient derivative with a decaying exponential. If m is positive, then we can use the results of the preceding paragraph to obtain

$$y(t + h) = y(t) + [y'(t)/m](1 - e^{-mh}) + \left[\mathcal{O}\left(\frac{1}{m^3}\right) \text{ or } \mathcal{O}(h^3)\right] \tag{5}$$

For $m > 0$, Eq. (5) yields a local approximation to Eq. (1) which will be useful either for a large m or a small h. If $m \le 0$, the above asymptotic results do not apply, but this is only true where we do not have decaying exponential behavior, so we merely use an appropriate polynomial (or nonstiff) explicit integration method in this case. It should be noted that the exponential approximation of Eq. (5), when evaluated for small h, yields $y(t + h) = y(t) + y'(t)h + y''(t)h^2/2 + \mathcal{O}(h^3)$ which is just a three-term Taylor expansion equivalent to a second-order explicit Runge-Kutta (RK2) approximation. Thus, when h is sufficiently small, the exponential approximation (which we only use for $m > 0$) will give results nearly the same as RK2. This suggests that the exponential and RK2 approximations can be blended nicely to form an explicit algorithm that is simple and very stable. Another key point regarding the practical implementation of this algorithm is that the same derivative evaluations required by RK2 can be used to estimate y'' by numerical differentiation, so that it is not necessary to analytically differentiate the equations for y' in order to compute the m values.

In regard to the stability of the new approach given here, the following can be said. Customary consideration of the usual prototype equation $y' = -\lambda y$, $\lambda > 0$, is not useful, since the new approach is necessarily unconditionally stable for this equation (which is, of course, extremely desirable). In fact, the exponential method integrates this equation exactly in a single step of any size. In general, a traditional stability analysis cannot be carried out, since at every step the integration of each variable would be accomplished by one of two different methods, exponential or nonexponential. A further complication is that the local m values depend on the approximate solution itself, and also on the method used to estimate them. However, numerical experiments on numerous challenging stiff problems reveal that the new approach generally has excellent stability properties.

Numerical Implementation

The exponential method defined in Eq. (5) was derived for a single ODE. Extension to a system of ODEs is straightforward. Thus, Eq. (5) can be written for any variable y_i (when $m_i > 0$) as

$$y_i(t + h) = y_i(t) + y_i'(t)\{1 - \exp[-m_i(t)h]\}/m_i(t) \tag{6}$$

where

$$m_i(t) + -y_i''(t)/y_i'(t)$$

To determine whether we have an exponentially decaying transient (stiff behavior), we first estimate $m_i(t) = -y_i''(t)/y_i'(t)$ at the beginning of the step by using numerical differentiation to approximate $y_i''(t)$. If $m_i > 0$, then we have local exponential decay for variable y_i and we use Eq. (6) to integrate the equation for that variable. If $m_i \leq 0$, then variable y_i is not exponentially decaying, and we use some other nonstiff explicit method. Here, we choose the second-order Runge-Kutta-trapezoidal as the nonstiff method. The sequence of calculations for one step of size h is then 1) calculate derivatives at start of step, $f_i[t,y(t)] \equiv d\text{old}_i$; 2) calculate Euler values $z(t+h) = y(t) + hf[t,y(t)]$ in loop; and 3) calculate the new derivatives $f_i[t+h, z(t+h)] \equiv d\text{new}_i$.

Note that steps 1–3 are required for the classical explicit second-order Runge-Kutta-trapezoidal method. In the following loop, the preceding calculations are used to automatically partition the variables into stiff and nonstiff groups and integrate each accordingly:

For $i = 1$ to N (number of variables)
  If $d\text{old}_i \neq 0$ then
    Calculate $m_i = (d\text{old}_i - d\text{new}_i)/(h \times d\text{old}_i)$
    If $m_i > 0$ then
      Calculate exponential approximation:
        $y_i(t+h) = y_i(t) + d\text{old}_i$
          $\times [1 - \exp(-m_i \times h)]/m_i$
    End If
  End If
  If $d\text{old}_i = 0$ or $m_i \leq 0$ then
    Calculate RK2 approximation:
      $y_i(t+h) = y_i(t) + (h/2) \times (d\text{old}_i + d\text{new}_i)$
  End If
Next i

Thus, at every step each variable is determined to be either stiff or nonstiff and is then integrated by the appropriate method.

Fixed-Step Integration

Although the full power of the new method will be achieved only by variable-step integration, it should be pointed out that the fixed-step mode for the new procedure is very simple to program. The new RK2/exponential (RK2/EXP) algorithm remains stable for substantially larger step sizes than the unaided RK2 and is capable of achieving modest accuracy at a much lower computational cost.

Variable-Step Integration

The efficiency of any stiff method (when used to integrate stiff problems) can only be realized with variable-step integration. Small step sizes are used only in the transient regions and then much larger step sizes are taken outside those regions. For variable-step integration with the new procedure, the local error for each method must be estimated. To accomplish this, the imbedding technique will be used (in which the local error is estimated as the difference between two methods of different order). For RK2, the local error is estimated as the difference between the solution obtained by RK2 and the readily available Euler value. This local error is, thus, $\mathcal{O}(h^2)$. To estimate the local error for the exponential method defined by Eq. (6), we need to obtain another approximation of the numerical solution by means of some other method. For this purpose, we approximate the solution by an exponential fitting over the step as follows:

$$w_i(t+h) = y_i(t) \exp[y_i'(t)h/y_i(t)] + \mathcal{O}(h^2) \tag{7}$$

Now, we estimate the local error for the exponential method as the difference between the solution $y_i(t+h)$ obtained by Eq. (6) and the estimate $w_i(t+h)$ obtained by Eq. (7). This local error is also $\mathcal{O}(h^2)$. Then, the following formula is used to adjust the step size:

$$h_{\text{new}} = \left(\frac{\text{relative error tolerance}}{\|\text{current relative local error}\|}\right)^{1/2} \times h_{\text{current}} \tag{8}$$

where

$$\|\text{current relative local error}\| = \max_{i=1}^{N}\left\{\frac{|\text{estimated local error for } i\text{th variable}|}{|y_i(t+h)| + \epsilon}\right\}$$

where ϵ is a positive parameter. If $|y_i(t+h)| > \epsilon$, then the relative error TOL is controlled. If $|y_i(t+h)| \leq \epsilon$, then the absolute error ϵTOL is controlled. The integration step size is then chosen as $h = \min\{h_{\text{new}}, 2h_{\text{current}}\}$. That is, the integration step size for the new step is not allowed to be more than twice the previous step size in order to reduce the number of failures and to allow for a smooth sequence of steps. The numerical results reported in the next section were obtained with ϵ set at 0.01. A complete Fortran program for the new RK2/EXP method is given elsewhere.[4]

Performance and Comparison

In this section, we compare the performance of the new RK2/EXP algorithm to the classical explicit second-order Runge-Kutta-trapezoidal method (referred to as RK2) on both stiff and nonstiff problems. The results reported for both methods were obtained with variable-step integration. The numerical experiments were performed on the Alliant FX-4 computer at the Department of Chemical and Nuclear Engineering, University of California, Santa Barbara, in double precision Fortran.

The numerical results are presented in terms of the following quantities:

$NTOL$ = desired local relative error for the nonexponential method (RK2); user-specified

$ETOL$ = desired local relative error for the exponential method; user-specified

%EXP = a measure of how much the exponential method was used as compared to RK2,

$$\left(\frac{\#\text{ EXP used}}{\#\text{ EXP used} + \#\text{ RK2 used}}\right) \times 100$$

RE = actual relative error in numerical solution

$$RE \equiv \frac{|\text{exact solution} - \text{compared solution}|}{|\text{exact solution}|}$$

Steps = total number of integration steps required

Time = computing time on the Alliant FX-4 in CPU seconds

Problem 1: Very Stiff ODE

$$y' = -1000\,y^2 + t^4, \qquad y(0) = 1$$

$$t_0 = 0 \quad \text{to} \quad t_f = 40$$

This single ODE describes a very stiff problem that is very challenging for the classical explicit methods (such as RK2). The degree of stiffness for this problem increases dramatically as the length of the integration interval increases. The problem was integrated out to $t = 40$ with an initial step size $h_0 = 0.01$. Table 1 shows the results for the unaided RK2 and the new RK2/EXP at $t = 40$ for various $NTOL$ and $ETOL$; the exact solution for y at $t = 40$ is $y(40) = 50.59641756$. For this problem, the unaided RK2 required an excessive amount of time (118 s) although a loose error tolerance (10^{-2}) was specified and gave a relative error of 2.01×10^{-3}. On the other hand, we can clearly see that the new RK2/EXP with $NTOL = ETOL = 10^{-2}$ required less than 6 s to perform the integration and gave a relative error of 2.9×10^{-5} (significantly

Table 1 Numerical results for RK2 and RK2/EXP for problem 1

Method	*NTOL*	*ETOL*	%EXP	*RE*	Steps	Time
RK2	10^{-2}	—	—	2.01×10^{-3}	674,703	118
RK2/EXP	10^{-2}	10^{-2}	99.5	2.9×10^{-5}	25,597	5.62
	10^{-2}	10^{-3}	99.8	5.4×10^{-6}	74,438	16.3

Table 2 Numerical results for RK2, RK2/EXP, and ITR for problem 2

Method	*NTOL*	*ETOL*	%EXP	*RE*	Steps	Time
RK2	10^{-2}	—	—	5.85×10^{-5}	3,735,224	941
RK2/EXP	10^{-2}	10^{-2}	53.8	1.43×10^{-2}	7,045	2.73
	10^{-2}	10^{-3}	77.4	1.07×10^{-3}	19,470	7.65
	10^{-2}	10^{-4}	79.7	2.05×10^{-5}	53,177	21.0
	10^{-3}	10^{-5}	81.6	1.49×10^{-6}	141,481	55.9
ITR	10^{-2}	—	—	8.88×10^{-4}	2,383	8.0
	10^{-3}	—	—	1.88×10^{-4}	2,978	10.1
	10^{-4}	—	—	7.08×10^{-5}	4,711	15.9
	10^{-5}	—	—	7.08×10^{-6}	14,884	50.2
	10^{-6}	—	—	7.08×10^{-7}	47,056	158

better accuracy at a dramatically lower cost). Hence, we can clearly see that the new RK2/EXP is substantially more efficient than RK2 for this problem.

Problem 2: Field-Noyes Chemical Oscillator[5]

$$y_1' = 77.27(y_2 - y_1y_2 + y_1 - 8.375 \times 10^{-6}y_1^2) \qquad y_1(0) = 4$$

$$y_2' = (-y_2 - y_1y_2 + y_3)/77.27 \qquad y_2(0) = 1.1$$

$$y_3' = 0.161(y_1 - y_3) \qquad y_3(0) = 4$$

$$t_0 = 0 \quad \text{to} \quad t_f = 300$$

This system is a famous model that represents a chemical oscillator (a chemical reaction with the concentrations of the three chemical species varying periodically in time) suggested by Field and Noyes.[5] This problem exhibits oscillatory behavior with a period of about 300 s and is very stiff. The problem was integrated out to $t = 300$ (as in Weimer and Clough[6]) with an initial step size $h_0 = 0.01$. Table 2 shows the results for the unaided RK2 and the new RK2/EXP at $t = 300$ for various *NTOL* and *ETOL*; relative errors reported are for the second variable at $t = 300$ whose exact solution is $y_2(300) = 1.290244186$. For this problem, the unaided RK2 required more than 3.7×10^6 steps and an excessive amount of computer time (941 CPU seconds) although a loose error tolerance was specified. On the other hand, we can clearly see that the new RK2/EXP takes substantially fewer steps for all values of error tolerances and is dramatically less expensive than RK2. For example, RK2/EXP required only 7045 steps and less than 3 s with $NTOL = ETOL = 10^{-2}$ to give a relative error of 1.43×10^{-2}. Notice, also, that RK2/EXP responds better to varying the tolerances than does RK2. For loose tolerances, RK2/EXP gives low accuracy; as the tolerances are tightened, the accuracy increases smoothly. Furthermore, RK2 gives high accuracy even for loose *NTOL* at a substantially higher cost. This is again due to stability controlling the step size for RK2. For this problem, the new method is most efficient when *NTOL* is loose (10^{-2}–10^{-3}) and *ETOL* is in the range 10^{-2}–10^{-4}. For example, it took RK2 about 941 s with $NTOL = 10^{-2}$ to give a relative error of 5.85×10^{-5}, whereas RK2/EXP needed only 21 s with $NTOL = 10^{-2}$ and $ETOL = 10^{-4}$ to give a relative error of 2.05×10^{-5} (better accuracy at a dramatically lower cost). Hence, the new RK2/EXP is substantially more efficient than RK2 for this problem when low or intermediate accuracy is desired.

Now that we have shown the new explicit RK2/EXP to be more efficient than the unaided explicit RK2, the question that presents itself is how does the new explicit scheme compare to the traditional implicit methods that have been found to be more efficient than the classical explicit methods for stiff problems. Here, we consider the well-known implicit trapezoidal rule (ITR). The results for the ITR are also presented in Table 2. These results were obtained with variable-step integration. The computer was used to generate the Jacobian matrix by numerical differentiation. (Recall that the fully explicit RK2/EXP does not require the evaluation of the Jacobian.) Note that the ITR requires at least 8 s to cover the integration interval, whereas the explicit RK2/EXP covers the integration interval in as little as 2.73 s for $NTOL = ETOL = 10^{-2}$ and provides a relative error of 1.43×10^{-2} (decent accuracy). Also, RK2/EXP responds better to varying tolerances than the ITR. RK2/EXP gives low accuracy for loose tolerances and increases the accuracy as the tolerances are tightened. On the other hand, the ITR gives high accuracy even for loose *TOL*. We can clearly see from the results of Table 2 that when low accuracy is desired, the explicit RK2/EXP is more efficient than the ITR. When high accuracy is desired, the ITR is more efficient than RK2/EXP. But considering the simplicity of the new explicit method and the ease of programing it as compared to the ITR (or any other implicit method), the RK2/EXP method is still competitive with the ITR for this problem.

Problem 3: Composite Heat Transfer Problem

Here, we solve a simple heat transfer problem that is of a type often occurring in aerospace applications. Suppose that a vehicle is exposed to very high temperatures, and insulation is required to keep the interior temperature below a prescribed value. A simple model can be postulated as follows. Let us assume that both the metallic body and the insulation are of unit length, and let us neglect the curvature of the body and use rectangular coordinates. Let the inner wall of the metallic body be held at, say, 100°F and the outer wall of the insulation be exposed to, say, 500°F. Let $x = 0$ coincide with the location of the inner wall of the metallic body, $x = 1$ coincide with the location of the outer wall of the insulation. Let region I denote the metallic body and region II denote the insulation. The temperature in region I is $u_I(x, \tau)$ and in region II is $u_{II}(x, \tau)$; α_I and α_{II} are the thermal diffusivities, and k_I and k_{II} are the thermal conductivities. Then, the following partial

Table 3 Numerical results for RK2, RK2/EXP, and ITR for problem 3

Method	*NTOL*	*ETOL*	%EXP	*RE*	Steps	Time
RK2	10^{-2}	—	—	1.82×10^{-3}	136.027	35.3
RK2/EXP	10^{-2}	10^{-2}	75.7	7.13×10^{-3}	44	0.023
	10^{-2}	10^{-3}	88.7	9.36×10^{-4}	102	0.052
ITR	10^{-2}	—	—	5.29×10^{-7}	38	0.11

differential equations (PDEs) describe the one-dimensional heat transfer problem:

$$\frac{\partial u_{\mathrm{I}}}{\partial \tau} = \alpha_{\mathrm{I}} \frac{\partial^2 u_{\mathrm{I}}}{\partial x^2}$$

$$\frac{\partial u_{\mathrm{II}}}{\partial \tau} = \alpha_{\mathrm{II}} \frac{\partial^2 u_{\mathrm{II}}}{\partial x^2}$$

If we rescale the time τ by defining $t = \alpha_{\mathrm{II}}\tau$, then the PDEs above become

$$\frac{\partial u_{\mathrm{II}}}{\partial t} = \frac{\alpha_{\mathrm{I}}}{\alpha_{\mathrm{II}}} \frac{\partial^2 u_{\mathrm{I}}}{\partial x^2}$$

$$\frac{\partial u_{\mathrm{II}}}{\partial t} = \frac{\partial^2 u_{\mathrm{II}}}{\partial x^2}$$

Initial conditions: at $t = 0$, $u_{\mathrm{I}} = u_{\mathrm{II}} = 100°\mathrm{F}$.

Boundary conditions:

1) At $x = 0$, $u_{\mathrm{I}} = 100°\mathrm{F}$.
2) At $x = 1$, $u_{\mathrm{I}} = u_{\mathrm{II}}$.
3) At $x = 1$, $k_{\mathrm{I}} = \partial u_{\mathrm{I}}/\partial x = k_{\mathrm{II}}(\partial u_{\mathrm{II}}/\partial x)$.
4) At $x = 2$, $u_{\mathrm{II}} = 500°\mathrm{F}$.

The PDE initial/boundary value problem above can be transformed into a system of first-order ODEs via the method of lines. To accomplish this, we discretize the spatial variable and use central differences to approximate the spatial derivatives. Let us place nodes in the spatial direction with an interval of $\Delta x = 0.5$, and let us define the following new variables:

$y_0 \equiv u_{\mathrm{I}}(x = 0,t) = 100°\mathrm{F}$
$y_1(t) \equiv u_{\mathrm{I}}(x = 0.5,t)$
$y_2(t) \equiv u_{\mathrm{I}}(x = 1,t) = u_{\mathrm{II}}(x = 1,t)$
$y_3(t) \equiv u_{\mathrm{II}}(x = 1.5,t)$
$y_4 \equiv u_{\mathrm{II}}(x = 2,t) = 500°\mathrm{F}$

After applying the method of lines, we end up with the following ODE system:

$$y_1' = 4\frac{\alpha_{\mathrm{I}}}{\alpha_{\mathrm{II}}}(y_2 - 2y_1 + 100) \qquad y_1(0) = 100$$

$$y_3' = 4(500 - 2y_3 + y_2) \qquad y_3(0) = 100$$

$$y_2' = 4\left(\frac{k_{\mathrm{I}}}{k_{\mathrm{II}}}y_1' + y_3'\right) \Big/ \left[3\left(\frac{k_{\mathrm{I}}}{k_{\mathrm{II}}} + 1\right)\right] \qquad y_2(0) = 100$$

$$t_0 = 0 \quad \text{to} \quad t_f = 1000$$

We chose both $(\alpha_{\mathrm{I}}/\alpha_{\mathrm{II}})$ and $(k_{\mathrm{I}}/k_{\mathrm{II}})$ equal to 100. The problem was integrated to $t = 1000$ with an initial step size of $h_0 = 0.01$. Table 3 presents the results for RK2, RK2/EXP, and the ITR; relative errors reported are for the second variable (which describes the temperature at the metal/insulation interface) at $t = 1000$ whose exact solution is $y_2(1000) = 107.9207921$. It is clear that the RK2/EXP method is substantially faster and more accurate than the unaided RK2. RK2/EXP is also faster than the ITR. Although the ITR gives higher accuracy than the RK2/EXP, it takes more time to perform the integration. RK2/EXP gives good accuracy for the specified loose tolerances. Furthermore, the explicit RK2/EXP method is easier to program and requires less computer memory than the ITR. Hence, RK2/EXP is far more efficient than RK2 and is competitive with ITR.

Table 4 Numerical results for RK2/EXP for problem 4

NTOL	*ETOL*	%EXP	*RE*	Steps	Time
10^{-2}	10^{-2}	0	3.22×10^{-2}	34	0.013
10^{-2}	10^{-6}	0	3.22×10^{-2}	34	0.013
10^{-3}	10^{-2}	0	2.91×10^{-3}	115	0.029
10^{-4}	10^{-2}	0	2.85×10^{-4}	371	0.078

Problem 4: Nonstiff Problem

$$y' = y^2 + t^2, \qquad y(0) = 1$$

$$t_0 = 0 \text{ to } t_f = 0.9$$

This is a purely nonstiff problem (i.e., the step size is controlled only by accuracy requirements without any stability restrictions). The purpose of this problem is to test whether the new RK2/EXP algorithm will recognize the nonstiff nature of the problem and integrate it by using the nonstiff method (RK2). The solution of this problem has a veritcal asymptote near $t = 1$; as $t \to 1$, $y \to \infty$. The problem was integrated out to $t = 0.9$ with an initial step size $h_0 = 0.01$. Table 4 presents the results for this problem; the exact solution for y at $t = 0.9$ is $y(0.9) = 14.30485943$. It can be clearly seen that the algorithm recognized the problem as nonstiff and automatically used the nonstiff RK2 method to integrate it (note that %EXP = 0 for all values of error tolerances indicating that the stiff exponential method was not used). This reveals the capability of the new RK2/EXP algorithm to efficiently integrate both stiff and nonstiff problems (a feature not usually encountered with general ODE codes).

Conclusions

A new explicit Runge-Kutta-trapezoidal/exponential algorithm that has much better stability properties than the classical explicit methods and which retains the essential simplicity in calculation and computer programming was developed in this paper. The procedure is capable of automatically determining whether the problem is stiff or nonstiff and of integrating either type efficiently. It is generally much more efficient than the unaided Runge-Kutta-trapezoidal (or the other explicit methods) for the integration of stiff systems of ordinary differential equations, especially when modest accuracy is desired. As shown here, the new explicit method performs extremely well in the integration of certain equations known to be stiff. In the integration of other equations, the new method performs in a qualitatively similar way, showing extremely high stability. When only modest accuracy is required, the procedure is much more efficient than traditional explicit methods. In general, it is not known whether a problem is stiff or nonstiff. Usually, a classical explicit nonstiff method (such as Runge-Kutta 4) is tried first. It is only when such a method starts to take very small step sizes that stiffness is suspected and an implicit stiff method is chosen instead. Since the Runge-Kutta-trapezoidal/exponential algorithm is usually able to efficiently integrate both stiff and nonstiff problems automatically, it would be better to use it first.

References

[1]Aiken, R. C. (ed.), *Stiff Computation,* Oxford University Press, New York, 1985, pp. 1, 7, 10, 16.

[2]Bender, C. M., and Orzag, S. A., *Advanced Mathematical Methods for Scientists and Engineers,* McGraw-Hill, New York, 1978, p. 263.

[3]Hanna, O. T., and Kapner, R. S., "Asymptotic Integration of Over-All Mass and Energy Balances for Nonisothermal Reaction Systems," *Industrial and Engineering Chemistry Fundamentals,* Vol. 6, No. 1, Feb. 1967, pp. 116–120.

[4]Ashour, S. S., "Investigation of New Methods for the Integration of Stiff Ordinary Differential Systems," Ph.D. Dissertation, Univ. of California, Santa Barbara, Santa Barbara, CA, Dec. 1989, pp. 178–186.

[5]Field, R. J., and Noyes, R. M., "Oscillations in Chemical Systems. IV. Limit Cycle Behavior in a Model of a Real Chemical Reaction," *Journal of Chemical Physics,* Vol. 60, No. 5, March 1, 1974, pp. 1877–1884.

[6]Weimer, A. W., and Clough, D. E., "A Critical Evaluation of the Semiimplicit Runge-Kutta Methods for Stiff Systems," *American Institute of Chemical Engineers Journal,* Vol. 25, No. 4, 1979, pp. 730–732.

[7]Lapidus, L., and Seinfeld, J. H., *Numerical Solution of Ordinary Differential Equations,* Academic Press, New York, 1971, p. 230.