MATHEMATICS OF COMPUTATION
VOLUME 36, NUMBER 153
JANUARY 1981

Quadrature Methods for Stiff Ordinary Differential Systems*

By A. Iserles

Abstract. The quadrature methods are based upon a substitution of an explicit A-stable first approximation into a generalized convolution formula. They are A-stable, explicit, and of arbitrarily high order.

The generalized convolution formula is derived and its order-raising properties examined. Two families of explicit A-stable first approximations are developed, generalizing results of Lawson and Nørsett. Various aspects of the numerical implementation are discussed.

Numerical results supplement the paper and exemplify the various merits and weaknesses of the quadrature methods.

1. Introduction. The purpose of this paper is to present a family of new methods for the numerical solution of stiff differential systems.

The methods of Nørsett [7], Jain [3], and Lawson [4] are A-stable, explicit, and of arbitrary order. Hence, they have the advantage, over implicit methods, that they do not require some sort of iteration. However, the methods of Nørsett-Jain-Lawson type suffer from one crucial deficiency: they are based on local approximation of the differential equations by a linear system. When the nonlinearity of the differential equations grows, the accuracy rapidly deteriorates and one is compelled to use excessively small step-lengths. This deterioration is accelerated if, as is usually the case, the Jacobian matrix is computed only once for several steps of the solution or if it is approximated by finite differences.

The quadrature methods are a generalization of the Nørsett-Jain-Lawson technique, or, in fact, of any explicit A-stable method. An explicit method is used to compute approximations to the solution for a wide range of points. These approximations are substituted into a generalized convolution formula, which is integrated by either Gaussian or Newton-Cotes quadrature.

These methods are derived by expanding a basic idea of Meister [6]. Meister advocates a computation of a polynomial approximation to the solution by some explicit method (for example explicit Runge-Kutta). This approximation is substituted into the convolution formula, which is integrated analytically (except that there is an approximation to the exponential of a matrix).

Our approach differs from Meister's on several points:

a. A generalized convolution formula is used, in order to cover the multiderivative case and to allow an increase in the order.

Received December 10, 1979; revised May 1, 1980.

1980 *Mathematics Subject Classification.* Primary 65L05.

*This is an expanded and revised version of *A-Stable Convoluted Predicator-Corrector Method*, DAMTP NA4, 1978.

0025-5718/81/0000-0012/$04.00

b. Instead of using polynomial approximation, we apply numerical integration to an arbitrary explicit first approximation. This is an advantage if high order is desired: using explicit Runge-Kutta for the construction of the polynomial approximation, one requires at least $m - 1$ function evaluations for order m, but, using a Gaussian quadrature method, one needs only $[(m + 1)/2] + 1$ function evaluations for order m.

c. By taking advantage of the special structure of matrices, which appear in both the Nørsett-Jain-Lawson type of methods and in the quadrature method, the amount of algebraic calculations is kept reasonably small. In fact, for large differential systems (where the overhead of algebraic calculations substantially matters), the additional algebraic expense of the quadrature, over the first approximation, is marginal.

In Section 2, we develop the generalized convolution formula and analyze its properties. In Section 3, we discuss the explicit first approximations. Section 4 is devoted to the computational aspects of the method. Finally, Section 5 gives numerical examples, which highlight the implementation of the method.

2. The Generalized Convolution Formula. Let the ordinary differential system

$$\mathbf{y}' = \mathbf{f}(t, \mathbf{y}), \quad \mathbf{y}(t_0) = \mathbf{y}_0 \in E^N \tag{1}$$

be given. For any N-by-N matrix A, the system (1) can be rewritten as

$$\mathbf{y}' = A\mathbf{y} + \mathbf{g}(t, \mathbf{y}), \quad \mathbf{y}(t_0) = \mathbf{y}_0,$$

where $\mathbf{g}(t, \mathbf{y}) = \mathbf{f}(t, \mathbf{y}) - A\mathbf{y}$. Hence, its solution is given by the convolution formula

$$\mathbf{y}(t) = \exp((t - t_0)A)\left\{\mathbf{y}_0 + \int_{t_0}^{t} \exp(-(s - t_0)A)\mathbf{g}(\mathbf{s}, \mathbf{y}(s))\, ds\right\}. \tag{2}$$

Normally, A is either the Jacobian matrix of (1) or an approximation to the Jacobian matrix. Although the convolution formula is important to the theory of differential equations and to control theory, it seems that the only explicit application to the numerical solution of ordinary systems is the one given by Meister [6].

As it is often easy to obtain higher derivatives of $\mathbf{y}$ from (1) by direct differentiation, $\mathbf{y}'' = (\partial/\partial t)\mathbf{f}(t, \mathbf{y}) + (\partial/\partial \mathbf{y})\mathbf{f}(t, \mathbf{y})\mathbf{y}'$ etc., it is advantageous to include higher derivatives in the convolution formula (2). This is done in

THEOREM 1. *Let* $\mathbf{F}_k(t, A) = (d^k/dt^k)(\exp(-(t - t_0)A)\mathbf{y}(t))$, $k = 0, 1, \ldots$. *Then, if* $\mathbf{y}$ *is in* C^n, *the following formula* (*generalized convolution formula*) *holds*:

$$\begin{aligned}\mathbf{y}(t) = \exp((t - t_0)A) &\sum_{k=0}^{n-1} \frac{1}{k!}(t - t_0)^k \mathbf{F}_k(t_0, A) \\ &+ \frac{1}{(n-1)!}\int_{t_0}^{t} (t - s)^{n-1}\mathbf{F}_n(s, A)\, ds.\end{aligned} \tag{3}$$

Proof. By a repeated integration by parts of the integral on the right, the formula (3) reduces to (2). □

If $\mathbf{y}(t)$ is the exact solution of (1) at t, then, by Leibnitz' rule,

$$\mathbf{F}_k(t, A) = \exp(-(t - t_0)A) \sum_{i=0}^{k} \binom{k}{i} (-1)^{k-i}\mathbf{y}^{(i)}(t), \qquad k = 0, \ldots, n,$$

where $\mathbf{y}^{(i)}(t) = (d^i/dt^i)\mathbf{y}(t)$, $i = 0, \ldots, n$. Hence, the generalized convolution formula (3) can be written in the form

$$\begin{aligned}\mathbf{y}(t) = {} & \exp((t - t_0)A) \sum_{k=0}^{n-1} \frac{1}{k!}(t - t_0)^k \sum_{i=0}^{k} \binom{k}{i} (-1)^{k-i} A^{k-i} \mathbf{y}_0^{(i)} \\ & + \frac{1}{(n-1)!} \int_{t_0}^{t} (t - s)^{n-1} \exp((t - s)A) \sum_{i=0}^{n} \binom{n}{i} (-1)^{n-i} A^{n-i} \mathbf{y}^{(i)}(s)\, ds.\end{aligned} \tag{3'}$$

Example. For $n = 1$, (3′) is identical to (2), the usual convolution formula. For $n = 2$, we obtain

$$\begin{aligned}\mathbf{y}(t) = {} & \exp((t - t_0)A)\{\mathbf{y}_0 + (t - t_0)(\mathbf{y}_0' - A\mathbf{y}_0)\} \\ & + \int_{t_0}^{t} \exp((t - s)A)(t - s)(A^2\mathbf{y}(s) - 2A\mathbf{y}'(s) + \mathbf{y}''(s))\, ds.\end{aligned}$$

In practice, of course, instead of using an exact exponential of a matrix in formula (3), one approximates it by a rational function. The integration is replaced by a quadrature formula.

The main advantage of the generalized convolution, which is that this formula raises the order of the initial approximation by n, is shown in the sequel.

The formula (3′) is used for the computation of $\mathbf{y}(t_1)$, $t_1 = t_0 + h$, in the following way:

We choose a quadrature formula with the nodes $t_0 + c_k h$ and weights w_k, $k = 1, \ldots, M$, and then evaluate some first approximation $\tilde{\mathbf{y}}_k$ to the solution of (1) at the quadrature nodes. This approximation is substituted into the integrand on the right of (3′), the exponentials are replaced by rational approximations, and finally the integral is approximated by the quadrature.

The derivatives of $\mathbf{y}$ at the quadrature nodes can be estimated by substitution of $\tilde{\mathbf{y}}_k$ into the formal differentials of (1):

$$\tilde{\mathbf{y}}_k' \approx \mathbf{f}(t_0 + c_k h, \tilde{\mathbf{y}}_k), \quad \tilde{\mathbf{y}}_k' \approx \frac{\partial}{\partial t}\mathbf{f}(t_0 + c_k h, \tilde{\mathbf{y}}_k) + \frac{\partial}{\partial \mathbf{y}}\mathbf{f}(t_0 + c_k h, \tilde{\mathbf{y}}_k)\tilde{\mathbf{y}}_k' \quad \text{etc.}$$

Observe that if $\tilde{\mathbf{y}}_k - \mathbf{y}(t_0 + c_k h) = O(h^{m+1})$, where $\mathbf{y}$ is the exact solution of (1) and f is in C^{n+1}, then the differentiability of the formal derivatives of (1) implies $\tilde{\mathbf{y}}_k^{(i)} - \mathbf{y}^{(i)}(t_0 + c_k h) = O(h^{m+1})$, $i = 0, \ldots, n$.

THEOREM 2. *Let the function f in* (1) *be in C^{n+1}, $t_1 = t_0 + h$ and $\{c_1, \ldots, c_M\}$, $\{w_1, \ldots, w_M\}$ nodes and weights, respectively, of a quadrature formula. Then, if*

(i) $\tilde{\mathbf{y}}_k^{(i)} - \mathbf{y}^{(i)}(t_0 + c_k h) = O(h^{m+1})$, $i = 0, \ldots, n$, $k = 1, \ldots, M$, *where* $\mathbf{y}$ *is the exact solution of* (1) *and* $\tilde{\mathbf{y}}_k^{(i)}$ *is the initial approximation to the ith derivative of* $\mathbf{y}$ *at the kth quadrature node;*

(ii) *the order of the quadrature is m;*

(iii) *the exponential which precedes the sum on the right of* (3′) *is replaced by an approximation R_1 of the order $n + m$;*

(iv) *the exponential in the integrand on the right of* (3′) *is replaced by an approximation R_2 of the order m*

then the order of the method is $n + m$, i.e.

$$\mathbf{y}^*(t_1) = R_1(hA)\sum_{k=0}^{n-1}\frac{1}{k!}h^k\sum_{i=0}^{k}\binom{k}{i}(-1)^{k-i}A^{k-i}\mathbf{y}_0^{(i)}$$

$$+\frac{1}{(n-1)!}h^n\sum_{k=1}^{M}\left\{w_k(1-c_n)^{n-1}R_2((1-c_k)hA)\sum_{i=0}^{n}\binom{n}{i}(-1)^{n-i}A^{n-i}\tilde{\mathbf{y}}_k^{(i)}\right\}; \tag{4}$$

$$\mathbf{y}^*(t_1) - \mathbf{y}(t_1) = O(h^{n+m+1}).$$

Proof. Let us assume first that we substitute on the right of (4) the exact solution y of the differential equation (1), obtaining an approximation $\mathbf{y}^{**}(t_1)$ on the left. Then

$$\mathbf{y}^{**}(t_1) - \mathbf{y}(t_1) = \{R_1(hA) - \exp(hA)\}\sum_{k=0}^{n-1}\frac{1}{k!}h^k\sum_{i=0}^{k}\binom{k}{i}(-1)^{k-i}A^{k-i}\mathbf{y}_0^{(i)}$$

$$+\frac{1}{(n-1)!}h^n\left\{\sum_{k=1}^{M}w_k(1-c_k)^{n-1}R_2((1-c_k)hA)\right.$$

$$\cdot\sum_{i=0}^{n}(-1)^{n-i}A^{n-i}\mathbf{y}^{(i)}(t_0+c_kh)$$

$$\left.-\int_0^1(1-s)^{n-1}\exp((1-s)hA)\sum_{i=0}^{n}(-1)^{n-i}A^{n-i}\mathbf{y}^{(i)}(s)\,ds\right\}.$$

But

$$R_1(hA) - \exp(hA) = O(h^{n+m+1});$$

$$R_2((1-c_k)hA) - \exp((1-c_k)hA) = O(h^{m+1}), \qquad k = 1, \ldots, M,$$

and the quadrature is of the order m. Hence,

$$\mathbf{y}^{**}(t_1) - \mathbf{y}(t_1) = O(h^{n+m+1}).$$

Finally,

$$\mathbf{y}^*(t_1) - \mathbf{y}^{**}(t_1) = \frac{1}{(n-1)!}h^n\sum_{k=1}^{M}w_k(1-c_k)^{n-1}R_2((1-c_k)hA)$$

$$\cdot\sum_{i=0}^{n}\binom{n}{i}(-1)^{n-i}A^{n-i}\{\tilde{\mathbf{y}}_k^{(i)} - \mathbf{y}^{(i)}(t_0+c_kh)\} = O(h^{n+m+1}),$$

according to condition (i). Therefore,

$$\mathbf{y}^*(t_1) - \mathbf{y}(t_1) = (\mathbf{y}^*(t_1) - \mathbf{y}^{**}(t_1)) + (\mathbf{y}^{**}(t_1) - \mathbf{y}(t_1)) = O(h^{n+m+1}),$$

and the proof follows. □

We call methods of this type *quadrature methods*.

3. First Approximation. The first approximation $\tilde{\mathbf{y}}$ is required at all the quadrature nodes in the interval $[t_0, t_0 + h]$. In order to minimize the computational effort, it is best to obtain $\tilde{\mathbf{y}}$ by an explicit method. It is demonstrated in the sequel that $\tilde{\mathbf{y}}$ can be computed for a whole range of points with just one evaluation of the system (1). In the next section we also show that just one *LU* factorization of a matrix (or related algebraic procedure) is required for the computation of the exponential approximations for the whole range of quadrature nodes.

There are two approaches, to be found in the literature, for constructing A-stable

explicit methods of arbitrary order. The first (Lawson, [4]) consists of a transformation of the system (1) and subsequent application of an explicit Runge-Kutta method. The second (Nørsett, [7]) gives linear multistep methods with matricial coefficients (this approach has been generalized by Jain [3], to include multistep methods with second derivatives).

For the sake of simplicity and in order to minimize the number of function evaluations, we are interested in linear multistep methods with matricial coefficients. If higher derivatives are available and the generalized convolution formula is used, we naturally are interested in multiderivative versions of these methods, which yield higher order. Finally, the functions are evaluated at the quadrature points, and so the mesh points cannot be assumed equispaced.

The *Lawson transformation* depends on the change of variable

$$\mathbf{z}(t) = \exp(-(t - t_0)A)\mathbf{y}(t),$$

where A is the Jacobian matrix. An explicit method is used to solve the differential equation that is satisfied by $\mathbf{z}$. Let

$$\sum_{i=0}^{q} \sum_{k=0}^{s} a_{ik}\mathbf{y}^{(k)}(t_{-i}) = \mathbf{y}(t_1) \tag{5}$$

be a zero-stable explicit multistep-multiderivative method of order m, where $t_1 > t_0 > t_{-1} > \cdots > t_{-q}$ and $\mathbf{y}^{(k)}$ denotes the kth derivative of $\mathbf{y}$.

Straightforward differentiation implies

$$\mathbf{z}^{(k)}(t) = \mathbf{F}_k(t, A, \mathbf{y}) = \exp(-(t - t_0)A) \sum_{i=0}^{k} (-1)^{k-j} A^{k-j}\mathbf{y}^{(j)}(t).$$

Hence, applying the approximation (5) to $\mathbf{z}$, instead of to $\mathbf{y}$, yields

$$\sum_{i=0}^{q} \sum_{k=0}^{s} a_{ik}\exp(-(t_{-i} - t_0)A) \sum_{j=0}^{k} (-1)^{k-j}\binom{k}{j} A^{k-j}\mathbf{y}^{(j)}(t_{-i})$$
$$= \exp(-(t_1 - t_0)A)\mathbf{y}(t_1),$$

or, after rearrangement,

$$\mathbf{y}(t_1) = \sum_{i=0}^{q} \exp((t_1 - t_{-i})A) \sum_{j=0}^{s} \left(\sum_{k=j}^{s} (-1)^{k-j}\binom{k}{i} a_{ik}A^{k-j} \right) \mathbf{y}^{(j)}(t_{-i}). \tag{6}$$

Instead of matrix exponentials we use appropriate approximations (this problem will be discussed later).

Evidently, the computation of $\mathbf{y}(t_1)$ requires just one evaluation of $\mathbf{y}^{(k)}$, $k = 1, \ldots, s$, at $t = t_0$ (the derivatives at t_{-i}, $i = 1, \ldots, q$, are known from previous steps). Furthermore, computation of $\mathbf{y}$ at several points does not require any extra function evaluations.

We call the second approach the *Hermite method*. For the q-step, $(s - 1)$th order derivative case let g_{ik}, $i = 0, \ldots, q$, $k = 0, \ldots, s - 1$, be the generalized Hermite interpolation polynomials with nodes at the previous integration points $t_0 > t_{-1} > \cdots > t_{-q}$, such that $\deg g_{ik} = (q + 1)s - 1$ and

$$\frac{d^p}{dt^p} g_{ik}(t_{-r}) = \delta_{i,r}\delta_{k,p}, \qquad r = 0, \ldots, q, p = 0, \ldots, s - 1,$$

where δ is the delta of Kronecker. Then it follows, as in [7] and [3], that the error of the approximation

$$\mathbf{y}'(t) - A\mathbf{y}(t) \approx \sum_{i=0}^{q} \sum_{k=0}^{s-1} g_{ik}(t)\left(\mathbf{y}^{(k+1)}(t_{-i}) - A\mathbf{y}^{(k)}(t_{-i})\right) \tag{7}$$

is of order $(q+1)s$.

Let A be the Jacobian matrix at $(t_0, \mathbf{y}_0)$. Multiplying (7) by $\exp(-tA)$ and integrating from t_0 to t_1 (the integration increases the order by one), we obtain

$$\mathbf{y}(t_1) = \exp((t_1 - t_0)A)\mathbf{y}(t_0) + \sum_{i=0}^{q} \sum_{k=0}^{s-1} H_{ik}\left(\mathbf{y}^{(k+1)}(t_{-i}) - A\mathbf{y}^{(k)}(t_{-i})\right), \tag{8}$$

where

$$H_{ik} = \int_{t_0}^{t_1} \exp((t_1 - p)A)g_{ik}(p)\,dp.$$

The integrals H_{ik} can be evaluated analytically. Afterwards the exponentials are replaced by suitable approximations.

Once again, the computation of $\mathbf{y}$ for a whole range of points requires just one evaluation of $\mathbf{y}^{(k)}$, $k = 1, \ldots, s$, at $t = t_0$.

Examples.

(i) $q = 0$, $s = 1$; order 1:

Lawson transformation: $\mathbf{y}(t_1) = \exp((t_1 - t_0)A)(\mathbf{y}(t_0) + (t_1 - t_0) - A\mathbf{y}(t_0))$;

Hermite method: $\mathbf{y}(t_1) = \mathbf{y}(t_0) + A^{-1}(\exp((t_1 - t_0)A) - I)\mathbf{y}'(t_0)$,

(ii) $q = 1$, $s = 2$; order 4:

Let $h = t_0 - t_{-1}$, $v = (t_1 - t_0)/h > 0$, $a = v - v^3 - \frac{1}{2}v^4$, $b = \frac{1}{2}v^2 + \frac{2}{3}v^3 + \frac{1}{4}v^4$.

Lawson transformation:

$$\begin{aligned}\mathbf{y}(t_1) = \exp(hvA)\Big\{&\left[(I - aA + bA^2)\mathbf{y}(t_0) + (aI - bA)\mathbf{y}'(t_0) + b\mathbf{y}''(t_0)\right] \\ &+ \exp(hA)\Big[\left(I - (v-a)A + \left(\tfrac{1}{2}v^2(1+v)^2 - b\right)A^2\right)\mathbf{y}(t_{-1}) \\ &\quad + \left((v-a)I - \left(\tfrac{1}{2}v^2(1+v)^2 - b\right)A\right)\mathbf{y}'(t_{-1}) \\ &\quad + \left(\tfrac{1}{2}v^2(1+v)^2 - b\right)\mathbf{y}''(t_{-1})\Big]\Big\};\end{aligned}$$

Hermite method:

$$\begin{aligned}\mathbf{y}(t_1) = \exp(hvA)\mathbf{y}(t_0) &+ H_{0,0}(\mathbf{y}'(t_0) - A\mathbf{y}(t_0)) + H_{0,1}(\mathbf{y}''(t_0) - A\mathbf{y}'(t_0)) \\ &+ H_{1,0}(\mathbf{y}'(t_{-1}) - A\mathbf{y}(t_{-1})) + H_{1,1}(\mathbf{y}''(t_{-1}) - A\mathbf{y}'(t_{-1})),\end{aligned}$$

where

$$\begin{aligned}H_{0,0} = 12\Big(&\left(I + \tfrac{1}{2}hA\right)\exp(hvA) \\ &- \left(I + \left(v + \tfrac{1}{2}\right)hA + \tfrac{1}{2}(v+1)h^2vA^2 + \left(\tfrac{1}{6}v + \tfrac{1}{4}\right)h^3vA^3\right)\Big)/h^3A^4;\end{aligned}$$

$$\begin{aligned}H_{0,1} = 6\Big(&\left(I + \tfrac{1}{3}hA\right)\exp(hvA) \\ &- \left(I + \left(v + \tfrac{1}{3}\right)hA + \left(\tfrac{1}{2}v + \tfrac{1}{3}\right)h^2vA + \tfrac{1}{6}(v+1)h^3v^2A^3\right)\Big)/h^2A^4;\end{aligned}$$

$$\begin{aligned}H_{1,0} = -12\Big(&\left(I + \tfrac{1}{2}hA - \tfrac{1}{12}h^3A^3\right)\exp(hvA) \\ &- \left(I + \left(v + \tfrac{1}{2}\right)hA + \tfrac{1}{2}(v+1)h^2vA^2 + \left(\tfrac{1}{6}v^3 + \tfrac{1}{4}v^2 - \tfrac{1}{12}\right)h^3A^3\right)\Big)/h^3A^4;\end{aligned}$$

$$\begin{aligned}H_{1,1} = 6\Big(&\left(I + \tfrac{2}{3}hA + \tfrac{1}{6}h^2A^2\right)\exp(hvA) \\ &- \left(I - \left(v + \tfrac{2}{3}\right)hA + \left(\tfrac{1}{2}v^2 + \tfrac{2}{3}v + \tfrac{1}{6}\right)h^2A^2 + \left(\tfrac{1}{6}v^2 + \tfrac{1}{3}v + \tfrac{1}{6}\right)h^3vA^3\right)\Big)/h^2A^4.\end{aligned}$$

In practical computation the exponential is replaced by some rational approximation. For example, if the first diagonal Padé approximation $R_{1,1}(x) = (1 - \frac{1}{2}x)^{-1}(1 + \frac{1}{2}x)$, $R_{1,1}(x) - \exp(x) = O(x^3)$, is used, the methods (i) are

$$\mathbf{y}(t_1) = \left(I - \tfrac{1}{2}(t_1 - t_0)A\right)^{-1}\left(I + \tfrac{1}{2}(t_1 - t_0)A\right)\left[\mathbf{y}(t_0) + (t_1 - t_0) - A\mathbf{y}(t_0)\right]$$

and

$$\mathbf{y}(t_1) = \mathbf{y}(t_0) + \left(I - \tfrac{1}{2}(t_1 - t_0)A\right)^{-1}(t_1 - t_0)\mathbf{y}'(t_0),$$

respectively.

4. Computational Aspects. The selection of an integration method in (3) is closely related to the selection of an exponential approximation. On the one hand, it is desirable to obtain an integration order as high as possible with few function evaluations. On the other hand, the number of algebraic operations, required by the evaluation of the exponential approximation at different points, should be small.

Because the nodes of integration of a closed Newton-Cotes formula are equispaced, one can take advantage of the multiplicative property of the exponential to reduce the volume of algebraic computations. However, Newton-Cotes integration is wasteful in terms of function evaluations.

As is well known, for a fixed order of integration, the greatest saving on function evaluations (i.e. the minimal number of integration nodes) is obtained for Gaussian quadrature (based on polynomials orthogonal in $[t_0, t_1]$ in respect to the weight function $(t_1 - s)^{n-1}$). According to Theorem 2 we are interested in integration of order m. Hence, it is possible to use either Gauss-Jacobi integration with q nodes, $q = [m/2] + 1$, or Gauss-Jacobi-Lobatto integration with $q + 1$ nodes, including the endpoints of the interval. These two methods are equivalent in terms of function evaluations and computation of exponential approximations.

The multiplicative property of the exponential is not helpful when Gaussian quadrature is used, because the nodes of the quadrature formula are not equispaced. In this case, however, much work can be saved by applying the multivalued exponential approximations [2]. The technique is to develop approximations to $\exp(qz)$ of the form $P(q, z)/Q(z)$. Because the denominator is independent of q, one only has to carry out once the major computational effort in the evaluation of an exponential approximation, which is to factorize the matrix of an algebraic linear system. The order of approximation of $P(q, z)/Q(z)$ for $0 < q < 1$ can be less than for $q = 1$, because of the conditions (iii) and (iv) of Theorem 2. It is shown in [2] that the Gauss-Jacobi-Lobatto integration has certain advantages in this context.

5. Numerical Examples. In this section, the performance of four quadrature methods is examined on three test problems. Their performance is compared with the analogous Lawson and Hermite methods. In order to provide a direct comparison, the tests were carried out with fixed step lengths and the same procedures for numerical algebra.

In the following, A denotes the Jacobian matrix of (1) at t_0, $\mathbf{y}_0 = \mathbf{y}(t_0)$, $\mathbf{y}_{1/2} = \mathbf{y}(t_0 + \frac{1}{2}h)$, $\mathbf{y}_1 = \mathbf{y}(t_0 + h)$, where $h = t_1 - t_0$. q and s have the same meanings as

in Section 3. $\exp(hA)$ is approximated by the second diagonal Padé approximation, $R = (I - \frac{1}{2}hA + \frac{1}{12}h^2A^2)^{-1}(I + \frac{1}{2}hA + \frac{1}{12}h^2A^2)$. In all cases $\tilde{\mathbf{y}}' = \mathbf{f}(t, \tilde{\mathbf{y}})$, $\tilde{\mathbf{y}}'' = (\partial/\partial t)\mathbf{f}(t, \tilde{\mathbf{y}}) + (\partial/\partial \mathbf{y})\mathbf{f}(t, \tilde{\mathbf{y}})\tilde{\mathbf{y}}'$.

QL_1: Quadrature of Lawson transformation with $q = 0$ (one step) and $s = 1$ (only the first derivative present). Trapezoidal integration is used and the order is two:

$$\tilde{\mathbf{y}}_1 = R(\mathbf{y}_0 + h(\mathbf{y}_0' - A\mathbf{y}_0)); \quad \mathbf{y}_1 = R\big(\mathbf{y}_0 + \tfrac{1}{2}h(\mathbf{y}_0' - A\mathbf{y}_0)\big) + \tfrac{1}{2}h(\tilde{\mathbf{y}}_1' - A\tilde{\mathbf{y}}_1).$$

QH_1: Quadrature of Hermite method with $q = 0$, $s = 1$. The details of integration and the order are the same as for QL_1:

$$\tilde{\mathbf{y}}_1 = \mathbf{y}_0 + A^{-1}(R - I)\mathbf{y}_0'; \quad \mathbf{y}_1 = R\big(\mathbf{y}_0 + \tfrac{1}{2}h(\mathbf{y}_0' - A\mathbf{y}_0)\big) + \tfrac{1}{2}h(\tilde{\mathbf{y}}_1' - A\tilde{\mathbf{y}}_1).$$

QL_2: Quadrature of Lawson transformation with $q = 0$, $s = 2$. Order 4 is obtained by two-point integration with nodes t_0 and $t_0 + \frac{1}{2}h$ and weights $(\frac{1}{6}, \frac{1}{3})$. $\exp(\frac{1}{2}hA)$ is approximated by the multivalued second diagonal Padé approximation [2],

$$\exp\big(\tfrac{1}{2}hA\big) \approx S = \big(I - \tfrac{1}{2}hA + \tfrac{1}{12}h^2A^2\big)^{-1}\big(I - \tfrac{1}{24}h^2A^2\big).$$

$$\tilde{\mathbf{y}}_{1/2} = S\big(\mathbf{y}_0 + \tfrac{1}{2}h(\mathbf{y}_0' - A\mathbf{y}_0) + \tfrac{1}{8}h^2(\mathbf{y}_0'' - 2A\mathbf{y}_0' + A^2\mathbf{y}_0)\big);$$

$$\mathbf{y}_1 = R\big(\mathbf{y}_0 + h(\mathbf{y}_0' - A\mathbf{y}_0) + \tfrac{1}{6}h^2(\mathbf{y}_0'' - 2A\mathbf{y}_0' + A^2\mathbf{y}_0)\big) + \tfrac{1}{3}S\big(\tilde{\mathbf{y}}_{1/2}'' - 2A\tilde{\mathbf{y}}_{1/2}' + A^2\tilde{\mathbf{y}}_{1/2}\big).$$

QH_2: Quadrature of Hermite method with $q = 0$, $s = 2$. The integration and exponential approximation are the same as for QL_2:

$$\tilde{\mathbf{y}}_{1/2} = \mathbf{y}_0 + \tfrac{1}{2}h\mathbf{y}_0' + A^{-2}\big(S - I - \tfrac{1}{2}hA\big)\mathbf{y}_0'';$$

$$\mathbf{y}_1 = R\big(\mathbf{y}_0 + h(\mathbf{y}_0' - A\mathbf{y}_0) + \tfrac{1}{6}h^2(\mathbf{y}_0'' - 2A\mathbf{y}_0' + A^2\mathbf{y}_0)\big) + \tfrac{1}{3}S\big(\tilde{\mathbf{y}}_{1/2}'' - 2A\tilde{\mathbf{y}}_{1/2}' + A^2\tilde{\mathbf{y}}_{1/2}\big).$$

These methods are compared with the corresponding Lawson and Hermite methods:

L_1: Lawson transformation with $q = 0$, $s = 1$ (order one):

$$\mathbf{y}_1 = R(\mathbf{y}_0 + h(\mathbf{y}_0' - A\mathbf{y}_0)).$$

H_1: Hermite method with $q = 0$, $s = 1$ (order one):

$$\mathbf{y}_1 = \mathbf{y}_0 + A^{-1}(R - I)\mathbf{y}_0'.$$

(L_1 and H_1 correspond to example (i) of Section 3, with the exponential replaced by the rational approximation R.)

L_2: Lawson transformation with $q = 0$, $s = 2$ (order two):

$$\mathbf{y}_1 = R\big(\mathbf{y}_0 + h(\mathbf{y}_0' - A\mathbf{y}_0) + \tfrac{1}{2}h^2(\mathbf{y}_0'' - 2A\mathbf{y}_0' + A^2\mathbf{y}_0)\big).$$

H_2: Hermite method with $q = 0$, $s = 2$ (order two):

$$\mathbf{y}_1 = \mathbf{y}_0 + h\mathbf{y}_0' + A^{-2}(R - I - hA)\mathbf{y}_0''.$$

It should be mentioned that all the eight methods are one-step (i.e. $q = 0$ in the formulation of Section 3), and so self-starting. Of course, in practical programming one can use, as well, methods with positive values of q, i.e. multistep.

The performances of these eight methods are compared on the following problems:

Problem 1.

$$y_1' = -\left(80 + \tfrac{1}{5}/(1+t)\right)y_1 - \left(40 - \tfrac{2}{5}/(1+t)\right)y_2;$$

$$y_2' = -\left(40 - \tfrac{2}{5}/(1+t)\right)y_1 - \left(20 + \tfrac{4}{5}/(1+t)\right)y_2;$$

$$y_1(0) = 0, \quad y_2(0) = 1.$$

This is a linear system with variable coefficients, whose analytic solution is

$$y_1(t) = 0.4(e^{-100t} - 1/(1+t)); \quad y_2(t) = 0.2(e^{-100t} + 4/(1+t)).$$

It is mildly stiff, with the eigenvalues -100 and $-1/(1+t)$.

The integration was carried out in the interval [0, 2], with a fixed step length. Table 1 gives the absolute error (in maximum norm) for different step lengths.

The table shows that the errors of all eight methods grow as h^p, where p is the corresponding order. This is in perfect agreement with the assumption that the local error is governed by the principal error term.

TABLE I

	$h = 0.025$	0.05	0.01	0.2
L_1	2.23×10^{-3}	4.46×10^{-3}	8.93×10^{-3}	1.74×10^{-2}
H_1	2.23×10^{-3}	4.46×10^{-3}	8.93×10^{-3}	1.74×10^{-2}
L_2	2.49×10^{-5}	1.00×10^{-4}	4.05×10^{-4}	2.14×10^{-3}
H_2	5.04×10^{-5}	2.06×10^{-4}	8.57×10^{-4}	4.21×10^{-3}
QL_1	1.25×10^{-5}	5.07×10^{-5}	2.08×10^{-4}	3.73×10^{-4}
QH_1	1.25×10^{-5}	5.07×10^{-5}	2.08×10^{-4}	3.73×10^{-4}
QL_2	1.98×10^{-9}	3.19×10^{-8}	5.14×10^{-7}	9.88×10^{-4}
QH_2	2.35×10^{-9}	3.80×10^{-8}	6.18×10^{-7}	9.88×10^{-4}

The following conclusions can be derived from Table I:

a. The quadrature device increases greatly the accuracy of the solution.

b. By comparing methods of the same order (L_2 and H_2 versus QL_1 and QH_1) it can be seen that the quadrature methods are marginally better. However, instead of, say, N evaluations of the functions y' and y'', which is required by L_2 and H_2, the quadrature methods require only $2N$ evaluations of y'. Hence, the quadrature methods avoid the programming effort, and perhaps an increase in computer time, for second derivatives.

c. The performances of L_1 and H_1 are identical, while L_2 performs better than H_2. The same behavior is shown by the quadrature methods.

Problem 2.

$$y_1' = -\tfrac{1}{5}\left[(4a+b)y_1 + (2a-2b)y_2\right] - 2ce^{qt}(2y_1 + y_2)^2/25;$$

$$y_2' = -\tfrac{1}{5}\left[(2a-2b)y_1 + (a+4b)y_2\right] - ce^{qt}(2y_1 + y_2)^2/25;$$

$$y_1(0) = 2 - d, \quad y_2(0) = 1 + 2d.$$

This problem was considered by Liniger [5]. The solution is

$$y_1(t) = 2e^{-at}/(1 + ct) - de^{-bt}; \quad y_2(t) = e^{-at}/(1 + ct) + 2de^{-bt}$$

and the eigenvalues are $-b$ and $-(a + 2c/(1 + ct))$. Following Liniger, the parameters were fixed as $a = 0.2$, $b = 200$, and $d = 0$. The problem was tested for the following values of the "nonlinearity coefficient": $c = 0$, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10. Table II gives the respective global absolute errors at $t = 2$ (in maximum norm), with step-length $h = 0.1$.

TABLE II

	$c = 0$	10^{-3}	10^{-2}	10^{-1}	1	10
L_1	1.19×10^{-10}	2.67×10^{-7}	2.55×10^{-5}	1.71×10^{-3}	1.62×10^{-2}	8.94×10^{-3}
H_1	1.19×10^{-10}	2.69×10^{-5}	2.60×10^{-4}	1.91×10^{-3}	3.61×10^{-3}	2.31×10^{-3}
L_2	1.19×10^{-10}	1.16×10^{-10}	8.25×10^{-9}	5.27×10^{-6}	3.75×10^{-4}	1.51×10^{-3}
H_2	1.19×10^{-10}	1.80×10^{-7}	1.73×10^{-6}	7.50×10^{-6}	3.32×10^{-4}	1.50×10^{-3}
QL_1	1.19×10^{-10}	1.29×10^{-10}	4.38×10^{-9}	2.52×10^{-6}	1.28×10^{-4}	5.45×10^{-4}
QH_1	1.19×10^{-10}	1.30×10^{-10}	4.66×10^{-9}	2.82×10^{-6}	2.13×10^{-4}	2.11×10^{-3}
QL_2	1.19×10^{-10}	1.25×10^{-10}	1.86×10^{-10}	2.32×10^{-9}	3.53×10^{-7}	5.04×10^{-5}
QH_2	1.19×10^{-10}	1.25×10^{-10}	1.86×10^{-10}	2.31×10^{-9}	6.48×10^{-7}	1.64×10^{-4}

This table illustrates the sensitivity of the methods to nonlinearity. Because Problem 2 is linear when $c = 0$, this column shows the error of the exponential approximation (which is the same for all eight methods). We note that:

a. The quadrature methods deteriorate more slowly when the problem is more nonlinear.

b. The quadrature methods of order 2, namely QL and QH, are somewhat better than L_2 and H_2, which are also of order two. One should remember also that the implementation of QL_1 and QH_1 does not require second derivatives.

c. The Lawson transformation performs consistently better for small positive values of c, although the L and H methods perform in the same manner when the nonlinearity is strong. This property, however, is not obtained by the quadrature methods.

Problem 3.

$$y_1' = -2999.8y_1 + 999.9y_2 + c(5y_1^2 - 2y_1y_2);$$
$$y_2' = -5999.8y_1 + 1999.7y_2 + c(6y_1^2 - y_2^2);$$
$$y_1(0) = 0, \quad y_2(0) = 1.$$

The problem is moderately stiff. For $c = 0$, the eigenvalues are -0.1 and -1000 and the stiffness ratio is 10^4. Its solution is

$$y_1(t) = Z_1(t) - Z_2(t); \quad y_2(t) = 3Z_1(t) - 2Z_2(t),$$

where

$$Z_1(t) = e^{-\frac{1}{10}t}/\left(1 + 10c\left(1 - e^{-\frac{1}{10}t}\right)\right);$$
$$Z_2(t) = e^{-1000t}/\left(1 + c\frac{1 - e^{-1000t}}{1000}\right).$$

Once again, the sensitivity to nonlinearity was tested, with $c = 0, -0.1, -1, -10$. Table III shows the respective relative errors at $t = 10$ (the relative errors in both components are equal), using the step-length sequence:

$$0 < t \leqslant 0.05 \Rightarrow h = 0.01 \quad \text{(boundary layer)};$$
$$0.05 < t \leqslant 0.5 \Rightarrow h = 0.025,$$
$$0.5 < t \leqslant 10 \Rightarrow h = 0.25.$$

TABLE III

	$c = 0$	-0.1	-1	-10
L_1	1.95×10^{-7}	9.27×10^{-3}	5.03×10^{-2}	8.37×10^{-2}
H_1	1.95×10^{-7}	1.79×10^{-4}	2.05×10^{-3}	6.59×10^{-3}
L_2	1.95×10^{-7}	1.12×10^{-5}	1.23×10^{-3}	5.81×10^{-3}
H_2	1.95×10^{-7}	1.79×10^{-4}	2.05×10^{-3}	6.59×10^{-3}
QL_1	1.95×10^{-7}	1.02×10^{-5}	2.09×10^{-4}	3.48×10^{-4}
QH_1	1.95×10^{-7}	4.88×10^{-6}	7.25×10^{-4}	4.22×10^{-3}
QL_2	1.95×10^{-7}	2.99×10^{-7}	1.01×10^{-6}	1.36×10^{-5}
QH_2	1.95×10^{-7}	2.90×10^{-7}	4.50×10^{-6}	8.33×10^{-5}

The improvement of the quadrature methods that is shown in Table III is consistent with Tables I and II. However, it is no longer true that L_1 is better than H_1, but L_2 is better than H_2. In the quadrature methods, the Lawson transformations and the Hermite methods perform similarly.

Inasmuch as it is possible to derive practical conclusions from only three test problems, the numerical results which are presented in this paper suggest:

(i) The quadrature device greatly improves the precision of a method. This is to be expected because the order increases.

(ii) The quadrature methods give a slight advantage over both Lawson and Hermite methods of the same order, as far as precision is concerned. However, their main advantage is that they avoid the need for some higher derivatives.

(iii) It is not possible to decide, on the basis of presented numerical evidence, whether the Lawson transformation or the Hermite method is more suitable.

It should be mentioned that, in order to incorporate the quadrature methods in a package for the numerical solution of stiff differential systems, it would be necessary to study devices for stepsize control and local error estimation. Among the many likely candidates for such a device, two seem the most promising:

(a) Error estimation technique of Zadunaisky type [8]. It consists of an additional solution, with the same stepsize, of a linear equation which is derived from (1) and whose close-form solution is known. The comparison of the numerical and the analytic solutions of this auxiliary equation yields an estimate of the local error in the solution of the system (1).

(b) Automatic step and order adjustment, as in the Gear method [1]. By taking advantage of the Nordsieck representation, Gear's method estimates the local error of backward difference multistep methods. Similar devices can be developed for the quadrature method using either Nordsieck representation or approximation to

the derivatives by polynomial interpolation. If the error in the approximation of $\exp(hA)$ is neglected, it is both easier and more natural to estimate the principal error term in the expansion of $\mathbf{y}(t) - \exp((t - t_0)A)\mathbf{y}(t_0)$ (i.e. in the evaluation of the integral in (2)) than in the expansion of $\mathbf{y}(t)$.

Acknowledgement. The author wishes to express his gratitude to Professor M. J. D. Powell of the University of Cambridge, whose helpful criticisms and remarks were instrumental to the composition of this paper.

King's College
University of Cambridge
Cambridge CB2 1ST, England

1. C. W. GEAR, "The automatic integration of stiff ordinary differential equations," *Information Processing* 68, Vol. 1, *Mathematics, Software*, North-Holland, Amsterdam, 1969, pp. 187–193.

2. A. ISERLES, *On Multivalued Exponential Approximation, SIAM J. Numer. Anal.* (To appear.)

3. R. K. JAIN, "Some A-stable methods for stiff ordinary differential equations," *Math. Comp.*, v. 26, 1972, pp. 71–77.

4. J. D. LAWSON, "Generalised Runge-Kutta processes for stable systems with large Lipschitz constants," *SIAM J. Numer. Anal.*, v. 4, 1967, pp. 372–380.

5. W. LINIGER, "High order A-stable averaging algorithms for stiff differential systems," *Numerical Methods for Differential Systems* (Eds., L. Lapidus & W. E. Schiesser), Academic Press, New York, 1976, pp. 1–23.

6. G. MEISTER, "Über die Integration von Differentialgleichungssystemen 1. Ordnung mit exponentiell angepassten numerischen Methoden," *Computing*, v. 13, 1974, pp. 327–352.

7. S. P. NØRSETT, *An A-Stable Modification of the Adams-Bashforth Methods*, (Conference on the Numerical Solution of Differential Equations, Dundee, Scotland, 1969), Springer, Berlin, 1969, pp. 214–219.

8. P. E. ZADUNAISKY, "On the estimation of errors propagated in the numerical integration of ordinary differential equations," *Numer. Math.*, v. 27, 1976, pp. 21–39.