

Stiffness Detection and Estimation of Dominant Spectrum with Explicit Runge-Kutta Methods

Kersti Ekeland * Brynjulf Owren† Eivor Øines‡

Department of Mathematical Sciences, NTNU, N-7034 Trondheim, Norway

Version November 14, 1997

Abstract

A new stiffness detection scheme based on explicit Runge-Kutta methods is proposed. It uses a Krylov subspace approximation to estimate the eigenvalues of the Jacobian of the differential system. The numerical examples indicate that this technique is a worthwhile alternative to other known stiffness detection schemes, especially when the systems are large and when it is desirable to know more about the spectrum of the Jacobian than just the spectral radius.

AMS Subject Classification: 65L05

Key Words: Explicit Runge-Kutta methods, Stiffness detection, Initial value problems.

1 Introduction

There are available a large number of computer codes which attempt to solve initial value problems in the general form

$$\dot{y} = f(t, y), \quad y(a) = y_0, \quad a \leq t \leq b, \quad (1)$$

where f is a nonlinear map, $f : \mathbb{R} \times \mathbb{R}^N \rightarrow \mathbb{R}^N$. Most modern codes use a stepsize selection strategy that tries in each step to choose the largest possible stepsize such that an estimate of the local truncation error is kept smaller than a user defined tolerance TOL. To proceed from t_{n-1} to t_n , one thus needs to estimate a stepsize which makes the error estimate at t_n approximately equal to TOL. If one succeeds in the sense that the estimated error at t_n is less than TOL the step is accepted and the integration can proceed. If the error estimate is larger than TOL, one must reject the step, choose a smaller stepsize, and try once again. This procedure is repeated until the step is accepted. It is quite common to attempt to make the

*Email: kersti@math.ntnu.no

†Email: bryn@math.ntnu.no

‡Email: eivor.oines@isi.no

error estimate equal to $\theta \cdot \text{TOL}$ where $\theta < 1$ is a “pessimist factor”. In this way, one hopes to avoid a large number of steps being rejected.

The numerical integration schemes used to solve (1) can be divided into two classes, explicit and implicit schemes. The former have a low computational cost per step and they are usually easy to implement. Explicit methods are usually preferred when (1) is nonstiff. We do not intend to give a precise definition of the notion of stiffness here. The term was first used by Curtiss and Hirschfelder in 1952 [1], but it has been refined several times, the texts [2, 7, 5] give a good account of the various definitions. It is perhaps true that a precise definition of stiffness is not crucial for practical purposes. When an initial value problem is stiff, one will typically observe that a code based on an explicit scheme will need to use extremely small stepsizes in order to compute a stable solution as opposed to one based on an implicit scheme. The natural question is then: Will the increase in stepsize obtained with an appropriate implicit scheme compensate for the additional cost involved per step? In other words, is the implicit scheme cheaper per unit time than the explicit scheme? Although this way of interpreting stiffness is somewhat intuitive, it will be fundamental to the ideas of this paper.

It is not necessarily an easy task to determine a priori from (1) whether the problem is stiff. Sometimes, the degree of stiffness can be established by carefully analysing the differential equation, taking into account the size of the tolerance to be used, the integration interval, the initial values, and possibly other effects that will influence the stiffness. But in many cases, it may be more useful to assess the degree of stiffness as the integration proceeds, that is, to apply a stiffness detection scheme. We shall focus on a new way of detecting stiffness when an explicit Runge-Kutta method is used to integrate (1). To begin with, we shall review some of the most popular stiffness detection schemes. Several of the most popular stiffness detection schemes were proposed by Shampine, see e.g. [10, 11, 12]. For an excellent account of these and other schemes, see Robertson [8].

Perhaps the simplest of all stiffness detection schemes is the method of *constant stepsize*. It is based on the experience that in many situations where a problem becomes stiff, the stepsize as determined by the stepsize selection formula hardly changes at all. This is certainly a cheap method, but it is not very reliable, and perhaps its most important use is for early indication of stiffness that can trigger the application of a more reliable test.

A reliable and not too costly technique is the method of the “Low-order comparison formula pair”. The idea is to use a comparison formula pair in addition to the basic formula pair during the integration process. The two error estimates obtained from the two formulas are used to assess the degree of stiffness in the problem. The comparison formula pair is designed to use the same stages as the basic formula, thus no extra function evaluations are required. The comparison formula has lower order of accuracy than the basic formula, hence when the problem is nonstiff, it is expected to yield a larger error estimate than the basic formula pair. However, if the comparison formula pair has a larger stability region than the basic formula pair, we would expect its error estimate to become smaller than that of the basic formula pair when the problem is stiff. In the basic formula pair, we expect in each step that the error estimate e_n satisfies

$$\|e_n\| \approx \theta \text{TOL}, \quad 0 \leq \theta \leq 1,$$

where θ is the pessimist factor mentioned above. Thus, if the error estimate of the comparison

formula pair, \tilde{e}_n satisfies $\|\tilde{e}_n\| \leq \|e_n\|$ it is an indication that the problem may be stiff.

The eigenvalues of the Jacobian J of f in (1) may frequently give useful information about the stiffness of the problem. An eigenvalue with a large negative real part can be an indication of stiffness. If such a situation occurs, it is not uncommon that this eigenvalue λ satisfies $|\lambda| = \rho(J)$ where $\rho(J)$ is the spectral radius of J . If f satisfies a Lipschitz condition with respect to its second argument, i.e.

$$\|f(x, u) - f(x, v)\| \leq L\|u - v\|$$

we have the condition

$$\rho(J) \leq \|J\| \leq L$$

hence, we can argue that the Lipschitz constant L may give some useful information about the stiffness of the problem. For the Fehlberg (4,5) pair, Robertson [8] suggests the local estimate for the Lipschitz constant given by

$$L_{est} = \frac{\|f(y_{n+1}) - f(y_{n+1} + Ke_{n+1})\|}{\|Ke_{n+1}\|}$$

where K is chosen such that $\|Ke_{n+1}\| = \sqrt{u}\|y_{n+1}\|$ and u is the machine precision.

In this paper we will consider the approximation of eigenvalues of the Jacobian by means of a modified version of the Arnoldi algorithm. The technique is based on the observation that the stages of an explicit Runge-Kutta method can be viewed as approximations to a basis for a Krylov subspace of the Jacobian. The use of Krylov subspace techniques in the numerical solution of ODEs is not a new idea. See for instance [5, pp 160-165] where such techniques are reviewed for systems of ODEs that can be partitioned into a stiff and a nonstiff part. Also, Shampine [12] has suggested an approach which is in some ways similar to Krylov subspace techniques for the purpose of stiffness detection. By combining the stages of an explicit Runge-Kutta methods, he obtains approximations to the powers $(hJ)^k(hf(x_n, y_n))$ and these are used to obtain a nonlinear version of the power method for computing eigenvalues.

After reviewing the original Arnoldi's method, we will present a modification of it that can be used on the stages of a Runge-Kutta method. It leads to approximations to the dominant eigenvalues of the Jacobian matrix associated with a system of ODEs and may conceivably be used for many purposes. But the numerical experiments we present are all applications to stiffness detection.

2 A review of Arnoldi's method

The material we use here is taken from [9]. Arnoldi's method is an orthogonal projection method onto the Krylov subspace $\kappa_m \subseteq \mathbb{C}^N$ generated by an initial vector v_1 and the vectors $Av_1, \dots, A^{m-1}v_1$. The dimension of κ_m is $\min\{m, \mu\}$ where μ is the degree of the minimal polynomial of A and v_1 . In what follows we shall always assume, without loss of generality, that $m \leq \mu$. Let A be an $N \times N$ complex matrix. The Arnoldi algorithm can be expressed as follows

Algorithm 2.1

```

Choose a vector  $v_1 \in \mathbb{C}^N$  of norm 1
for  $j = 1, \dots, m$  do
   $w := Av_j$ 
  for  $i = 1, \dots, j$  do
     $h_{ij} := \langle w, v_i \rangle$ 
     $w := w - h_{ij}v_i$ 
  end
   $h_{j+1,j} := \|w\|$ 
   $v_{j+1} := w/h_{j+1,j}$ 
end

```

The inner product above and in what follows is defined by $\langle x, y \rangle = y^H x$ for any $x, y \in \mathbb{C}^N$. Also, by convention, the norm we refer to will always be the 2-norm $\|x\| = \sqrt{\langle x, x \rangle}$ for any $x \in \mathbb{C}^N$.

The vectors v_1, \dots, v_m form an orthonormal basis for κ_m . By letting V_m be the $N \times m$ matrix whose columns are v_1, \dots, v_m , and H_m the $m \times m$ upper Hessenberg matrix whose elements are defined in the algorithm, we can write

$$\begin{aligned} AV_m &= V_m H_m + h_{m+1,m} v_{m+1} e_m^H \\ V_m^H AV_m &= H_m \end{aligned} \quad (2)$$

where e_m is the m th canonical unit vector in \mathbb{C}^N .

We shall be interested in approximating the eigenvalue problem

$$Au = \lambda u, \quad u \in \mathbb{C}^N, \quad \lambda \in \mathbb{C}.$$

Our intention is to find an approximate eigenpair $\tilde{u} \in \kappa_m$, $\tilde{\lambda} \in \mathbb{C}$ such that the Galerkin condition

$$\langle A\tilde{u} - \tilde{\lambda}\tilde{u}, v \rangle = 0, \quad \forall v \in \kappa_m$$

is satisfied. Since $\tilde{u} \in \kappa_m$, we have $\tilde{u} = V_m y$ for some $y \in \mathbb{C}^m$ and the Galerkin condition can be rewritten as

$$H_m y = \tilde{\lambda} y,$$

where H_m is given by (2). Thus, approximating the eigenvalue λ of A by means of the Galerkin condition is equivalent to finding an eigenvalue of the upper Hessenberg matrix H_m obtained from the Arnoldi algorithm. Observe also that since modern versions of the QR method for finding eigenvalues of a matrix usually perform a preprocessing by computing a similarity transform of upper Hessenberg type, hence this step can be omitted if the QR algorithm is used to obtain the eigenvalues of H_m . It should be noted that in this approach we only find approximations to m of the N eigenvalues of A .

To assess the convergence properties of this approach, we shall consider a result from Saad [9]. Assume now that A is diagonalizable and that the initial vector has the eigenvector expansion

$v_1 = \sum_{k=1}^N \alpha_k u_k$. The distance in 2-norm between the eigenvector u_1 and its projection onto κ_m is bounded as follows:

$$\|(I - P_m)u_1\| \leq \xi_1 \epsilon_1^{(m)}, \quad \xi_1 = \sum_{k=2}^N \frac{|\alpha_k|}{|\alpha_1|}$$

For $m < N$ we then have the following result: There exists m eigenvalues of A , labelled $\lambda_2, \dots, \lambda_{m+1}$ such that

$$\epsilon_1^{(m)} = \left(\sum_{j=2}^{m+1} \prod_{k=2, k \neq j}^{m+1} \frac{|\lambda_k - \lambda_1|}{|\lambda_k - \lambda_j|} \right)^{-1} \quad (3)$$

To interpret this expression, let the eigenvalue λ_1 belong to the outermost part of the spectrum of A . We expect that the numbers $|\lambda_k - \lambda_1|$ will then be larger than the numbers $|\lambda_k - \lambda_j|$ of the denominator. Therefore, many of the products in (3) will be large, and the inverse of their sum will be small. From this we may conclude that if an eigenvalue is much larger in absolute value than most other eigenvalues, we expect the corresponding eigenvector to be well approximated by the above algorithm. We have found no result which relates directly the eigenvalue λ_1 to the approximate one $\tilde{\lambda}_1$ corresponding to $\tilde{u}_1 = P_m u_1$. However, assume that $\|u_1\| = 1$ and that for some m , we have $\|u_1 - \tilde{u}_1\| < 1$. It follows from the Galerkin condition that

$$\langle A(u_1 - \tilde{u}_1), v \rangle = \lambda_1 \langle u_1, v \rangle - \tilde{\lambda}_1 \langle \tilde{u}_1, v \rangle = (\lambda_1 - \tilde{\lambda}_1) \langle u_1, v \rangle + \tilde{\lambda}_1 \langle u_1 - \tilde{u}_1, v \rangle$$

for any $v \in \kappa_m$. Thus, we find that

$$|\lambda_1 - \tilde{\lambda}_1| |\langle u_1, v \rangle| \leq (|\tilde{\lambda}_1| + \|A\|) \|u_1 - \tilde{u}_1\| \|v\| \leq 2\|A\| \|u_1 - \tilde{u}_1\| \|v\| \quad \forall v \in \kappa_m$$

where we have used that $|\tilde{\lambda}_1| \leq \|H_m\| \leq \|A\|$. We now choose $v = \tilde{u}_1 \in \kappa_m$ and compute

$$1 = \langle u_1, u_1 \rangle \leq |\langle u_1, \tilde{u}_1 \rangle| + |\langle u_1, u_1 - \tilde{u}_1 \rangle| \leq |\langle u_1, \tilde{u}_1 \rangle| + \|u_1\| \|u_1 - \tilde{u}_1\|$$

to deduce that

$$|\langle u_1, \tilde{u}_1 \rangle| \geq 1 - \|u_1 - \tilde{u}_1\|$$

Moreover $\|v\| = \|\tilde{u}_1\| \leq 1 + \|u_1 - \tilde{u}_1\|$ so we finally find that

$$|\lambda_1 - \tilde{\lambda}_1| \leq 2\|A\| \frac{1 + \|u_1 - \tilde{u}_1\|}{1 - \|u_1 - \tilde{u}_1\|} \cdot \|u_1 - \tilde{u}_1\|$$

which relates the difference in eigenvalues to the difference in eigenvectors.

3 Stiffness detection with Arnoldi's method

To begin with, we consider the linear problem

$$\dot{y} = My + b, \quad M \in \mathbb{C}^{N \times N}, \quad y, b \in \mathbb{C}^N. \quad (4)$$

Explicit Runge-Kutta schemes for (1) have the following form

$$\begin{aligned} k_r &= f(t_n + c_r h, y_n + h \sum_{j=1}^{r-1} a_{rj} k_j), \quad r = 1, \dots, s \\ y_{n+1} &= y_n + h \sum_{r=1}^s b_r k_r \end{aligned}$$

Applied to (4) the stage values take the form

$$\begin{aligned} k_1 &= M y_n + b \\ k_r &= k_1 + \sum_{j=1}^{r-1} a_{rj} \tilde{M} k_j, \quad \text{where } \tilde{M} = hM \end{aligned} \quad (5)$$

Thus, it follows that under the condition $a_{r,r-1} \neq 0$, $r = 2, \dots, s$, the stage value k_r belongs to the Krylov subspace $\kappa_r(k_1, \tilde{M})$. In fact, the condition $a_{r,r-1} \neq 0$ is not at all crucial. Without loss of generality, we assume that $a_{21} \neq 0$. Consider then the sequence $2 = i_1 < i_2 < \dots$, where $i_m = \min\{i : a_{i,m} \neq 0\}$. We then have $k_{i_m} \in \kappa_m(k_1, \tilde{M})$. It is quite rare that Runge-Kutta methods have $a_{r,r-1} = 0$ for the stages we wish to use, and to avoid cluttered notation we shall subsequently always assume that $a_{r,r-1} \neq 0$.

We may also observe that the dimension of the available Krylov subspace is not necessarily limited to s . We can proceed into the next step. Now, using step indices we get

$$\begin{aligned} k_1^{n+1} &= k_1^n + \sum_{r=1}^s b_r \tilde{M} k_r^n \in \kappa_{s+1}(k_1^n, \tilde{M}) \\ k_r^{n+1} &= k_1^{n+1} + \theta_{n+1} \sum_{j=1}^{r-1} a_{rj} \tilde{M} k_j^{n+1} \in \kappa_{s+r}(k_1, \tilde{M}) \quad \text{where } \theta_{n+1} = h_{n+1}/h_n \end{aligned}$$

Again, for ease of notation and since our experience shows that it is not necessary to use a Krylov subspace of dimension larger than $s - 1$, we shall only use stages from one step at a time.

The introduction of \tilde{M} suggests that we wish to estimate the eigenvalues of hM rather than those of M . If y_n and w_n are two numerical approximations to $y(t_n)$, we have the iteration

$$y_{n+1} - w_{n+1} = p(\tilde{M})(y_n - w_n)$$

where $p(z)$ is the stability polynomial of the Runge-Kutta method. Hence it makes sense to consider the eigenvalues of \tilde{M} .

In Arnoldi's algorithm, the orthonormal basis $\{v_1, \dots, v_s\}$ is computed by applying a Gram-Schmidt type of orthogonalization procedure repeatedly to the vectors $\tilde{M}v_i$, $i = 1, \dots, s - 1$. Our situation is somewhat different, since the basis which is available is of a different form.

We begin by applying a QR -factorization to the stages k_1, \dots, k_s , keeping in mind that $k_r \in \kappa_r(k_1, \tilde{M})$. We thus compute

$$\begin{aligned} w_1 &:= k_1, & v_1 &:= w_1 / \|w_1\| \\ w_r &:= k_r - \sum_{j=1}^{r-1} \langle k_r, v_j \rangle v_j, & v_r &:= w_r / \|w_r\|, \quad r = 2, \dots, s \end{aligned}$$

The matrix V_s with columns v_1, \dots, v_s can be used to form

$$H_s := V_s^H \tilde{M} V_s$$

of which the eigenvalues can be computed. However, this approach only works when \tilde{M} is known explicitly, it would be more convenient to use only the stage values k_1, \dots, k_s such that the approach can be generalized to problems of the form (1). At least for autonomous problems, the k_i 's will serve as an approximate basis to a Krylov subspace of the Jacobian J . The cost of not using \tilde{M} explicitly is that we only obtain dimension $s-1$ of the Krylov subspace we compute. Let K_{s-1} be the matrix with columns k_1, \dots, k_{s-1} . Recall that $V_{s-1} V_{s-1}^H$ is a projection onto κ_{s-1} , thus

$$V_{s-1} V_{s-1}^H \cdot K_{s-1} = K_{s-1} \quad (6)$$

and therefore

$$\tilde{M} \cdot V_{s-1} = \tilde{M} \cdot K_{s-1} \cdot (V_{s-1}^H K_{s-1})^{-1} \quad (7)$$

We let $K_{2\dots s}$ be the matrix with columns k_2, \dots, k_s . The expression (5) can then be written in matrix form as

$$K_{2\dots s} = k_1 \mathbb{1}^H + \tilde{M} K_{s-1} A_1 \quad \text{where} \quad A_1 = \begin{bmatrix} a_{21} & a_{31} & \cdots & a_{s,1} \\ 0 & a_{32} & \cdots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & a_{s,s-1} \end{bmatrix}$$

and $\mathbb{1}$ is the $(s-1)$ -vector of ones, $(1, \dots, 1)^H$. By combining this expression with (7), we obtain

$$H_{s-1} = V_{s-1}^H \cdot (K_{2\dots s} - k_1 \mathbb{1}^H) \cdot A_1^{-1} \cdot (V_{s-1}^H K_{s-1})^{-1} \quad (8)$$

We suggest the following algorithm for the computation of H_{s-1} as given in (8).

Algorithm 3.1

Compute the nonzero elements of A_1^{-1} , α_{ij} , $i \leq j \leq s$, before integration begins.

Set $\beta_{11} := \|k_1\|$, and $v_1 := k_1/\beta_{11}$

for $j = 1, 2, \dots, s-1$ **do**

for $i = 1, \dots, j$ **do**

$\beta_{i,j+1} := \langle k_{j+1}, v_i \rangle$

end

$w := k_{j+1} - \sum_{i=1}^j \beta_{i,j+1} v_i$

$\beta_{j+1,j+1} := \|w\|$

$v_{j+1} := w/\beta_{j+1,j+1}$

for $i = 1, \dots, j+1$ **do**

$\ell = \max\{1, i-1\}$

$h_{ij} := \left(-\sum_{k=\ell}^{j-1} h_{ik} \beta_{kj} + \sum_{k=\ell}^j \beta_{i,k+1} \alpha_{kj} - \delta_{i1} \beta_{11} \sum_{k=1}^j \alpha_{kj} \right) / \beta_{jj}$

end

end

where the standard convention, $\delta_{ij} = 0$, $i \neq j$ and $\delta_{ii} = 1$ is used. Also sums where the lower index exceeds the upper index is taken as zero.

The resulting matrix $H_{s-1} = (h_{ij})_{i,j=1}^{s-1}$ is used for approximating the dominant eigenvalues of the matrix $\tilde{M} = hM$.

Even if the degree of the minimal polynomial of (\tilde{M}, k_1) is greater than $s - 1$, it may happen that the process must be terminated for $j < s - 1$. One should therefore test in each run through the j -loop of the algorithm whether $\|w\| < \delta \|k_{j+1}\|$ where δ is some small user defined tolerance. To perform such a test it is useful to have made the observation that

$$\|w\| = |\beta_{j+1,j+1}| \quad \text{and} \quad \|k_{j+1}\| = \left(\sum_{i=1}^{j+1} \beta_{i,j+1}^2 \right)^{1/2}$$

The above algorithm was developed based on the test problem (4). However, since it only uses the stages k_r , we may also apply the algorithm to (1) and we expect the resulting eigenvalues to approximate the outer part of the spectrum of the Jacobian of f .

Computational cost. Algorithm 3.1 takes

$$\left((s^2 + s - 1) \cdot N + \frac{1}{3}s^3 + \frac{5}{3}s - 4 \right) \text{ flops,} \quad s \geq 2$$

where 1 flop is approximately one addition and one multiplication. Also there are

$$\left((s - 1) \cdot N + \frac{1}{2}s^2 + \frac{1}{2}s + 1 \right) \text{ divisions,} \quad \text{and } s \text{ square roots}$$

In addition to this there is the cost of computing the eigenvalues of H_{s-1} by a suitable method, for instance the double-shifted QR algorithm. According to Golub and Van Loan [4] this takes approximately $8(s-1)^3$ flops. In conclusion, the dependence on N here is linear. In contrast, if for instance (4) is solved by an explicit Runge-Kutta method where M is a full matrix, the cost of taking a step is approximately

$$\left(s \cdot N^2 + \left(\frac{1}{2}s^2 + \frac{5}{2}s \right) \cdot N \right) \text{ flops}$$

Testing for stiffness. It is usually an indication of stiffness when eigenvalues of hJ with negative real part are located near the boundary of the stability region of the method. J is the Jacobian of f . The stability region of a method is the set of complex numbers

$$S_p = \{z \in \mathbb{C} : |p(z)| \leq 1\}$$

The stability polynomial $p(z)$ of the method can be defined in terms of the Runge-Kutta coefficients as follows: Let A be the $s \times s$ matrix with ij -element a_{ij} if $i > j$ and 0 otherwise. Let b be the s -vector $(b_1, \dots, b_s)^T$. Then

$$p(z) = 1 + zb^T(I - zA)^{-1}\mathbb{1}, \quad \mathbb{1} = (1, \dots, 1)^T \in \mathbb{R}^s$$

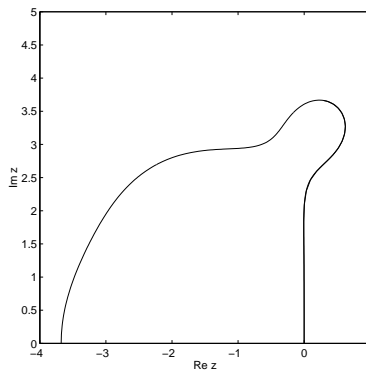


Figure 1: Stability region of the RKF45 method. The region is symmetric around the real axis. $p(z) = 1 + z + \frac{1}{2}z^2 + \frac{1}{6}z^3 + \frac{1}{24}z^4 + \frac{1}{120}z^5 + \frac{1}{2080}z^6$

For the most popular Runge-Kutta methods S_p looks approximately like a semicircle in the left half plane, centered at the origin, see for instance Figure 1.

For a thorough account of a possible detection strategy, see for instance Ekeland and Øines [3]. Their idea is to use the criteria of Dekker and Verwer [2, p.9] for linear problems to determine if the problem is stiff. Ultimately one then needs to determine whether an eigenvalue with negative real part exists and is located near the boundary of S_p . The closeness to the boundary can be assessed by computing $|p(\lambda)|$.

4 Numerical results

The following examples were run on a UNIX work station using Matlab. The eigenvalues of the upper Hessenberg matrix H_{s-1} were obtained with the use of the `eig` function in Matlab. In all examples we present here, we have used the RKF4(5) method due to Fehlberg. However, tests done with the 5(4) method by Dormand and Prince show similar results and we have found no reason to believe that the performance of the new stiffness detection scheme depends strongly on the choice of integration method.

The reader may notice that in some of the examples, the number of eigenvalues of the matrix H resulting from the Arnoldi process is less than $s - 1$. In fact, we have added a stopping

criterion to Algorithm 3.1, causing the modified Arnoldi process to terminate whenever it happens for some j that

$$\|w\| < \delta \|k_{j+1}\|.$$

The value of δ was set experimentally to $\approx 10^{-6}$. The consequence of not doing this, is that rounding error may cause the introduction of large “false” eigenvalues of the matrix H .

Example 1. Nonlinear problem with real eigenvalues. The reaction-diffusion PDE

$$u_t = u_{xx} + u(1 - u), \quad u(0, t) = u(1, t) = 1, \quad u(x, 0) = x(1 - x)$$

has an attractive fixed point at $u(x, t) \equiv 1$. We apply the method of lines by using centered differences for the u_{xx} term and obtain in the obvious way an ODE system of the type

$$U' = AU + F(U) + g, \quad U \in \mathbb{R}^N \tag{9}$$

where A is a tridiagonal matrix, $(F(U))_k = U_k(1 - U_k)$ and g is a constant vector that accounts for the boundary conditions. In Figure 2 we show the result of applying the new stiffness detection technique to (9) with $N = 39$. The plot shows the eigenvalues of the scaled Jacobian $hJ = hA + hF'(U)$ as dots, and the approximating eigenvalues resulting from the Arnoldi process are shown as circles. To improve visibility, the eigenvalues are only plotted at certain time steps. Notice how the outermost part of the spectrum of hJ is well approximated, especially after some time steps.

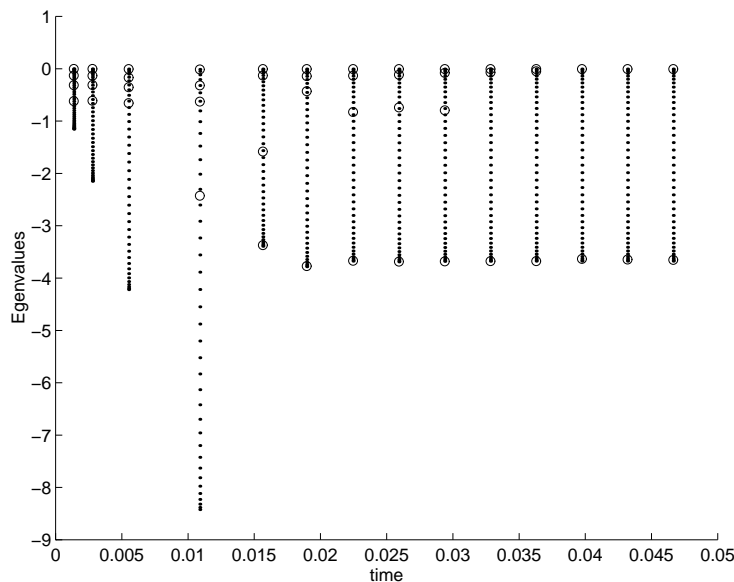


Figure 2: Exact (dots) and approximate eigenvalues (circles) for a discretized PDE

Example 2. In many numerical experiments, we have found that the comparison formula performs well, it is usually capable of detecting stiffness fast and in a reliable way. It does not give as detailed information of the problem as the new technique, but it is less computationally expensive, and therefore if the issue is only to decide between “stiff” and “nonstiff”, it may

be preferable to the new technique. We have, however, found cases where the new technique detects stiffness correctly and much faster than the comparison formula. Consider again the PDE of the previous example, but now we add the term $100u$, i.e. we consider

$$u_t = u_{xx} + u(1 - u) + 100u$$

with the same initial/boundary data and spatial discretisation as in Example 1. The extra term causes “locally” a right translation of the spectrum of hJ . For this reason, hJ will have small positive eigenvalues in the time range $0 \leq t \leq 0.062$. There will, however, still be a stable fixed point solution $u^*(x)$ which attracts the chosen initial function, u^* being close to the constant value 101 except for the thin layers near $x = 0$ and $x = 1$ where it drops down to the boundary value 1. Figure 3 shows the times when the two techniques detect stiffness. The comparison formula (lower part) detects stiffness when the two lines crosses each other, i.e. when the error estimate of the comparison formula becomes less than that of the basic formula. The Arnoldi based technique will detect the problem as stiff when $|p(h\lambda)| \approx 1$ for the eigenvalue λ with the largest negative real part. In the upper part of Figure 3 we show $|p(h\lambda)|$ as time proceeds, one can for instance let the detection technique flag “stiff” when $|p(h\lambda)|$ has stayed between the two horizontal dashed lines for a certain number of time steps.

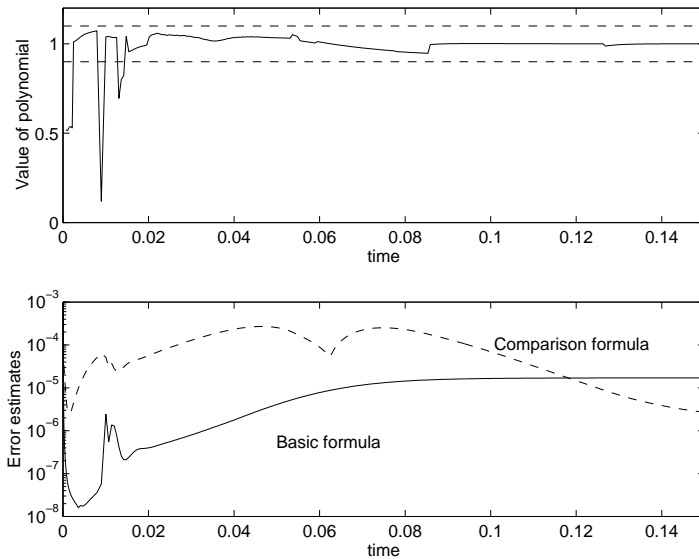


Figure 3: New technique compared to comparison formula for a discretized PDE.

Example 3. Complex eigenvalues. This example is artificial. Consider the linear problem

$$y' = Ay, \quad A \in \mathbb{R}^{32 \times 32}$$

where A has 16 complex conjugate pairs of eigenvalues, more precisely, they are of the form $\lambda_{2k-1} = a_k + i\sqrt{-a_k}$, $\lambda_{2k} = a_k - i\sqrt{-a_k}$, $a_k = -k$, $k = 1, \dots, 16$. A is constructed by computing a random similarity transform of the blockdiagonal matrix with 2×2 blocks

$$D_k = \begin{bmatrix} a_k & \sqrt{-a_k} \\ -\sqrt{-a_k} & a_k \end{bmatrix}, \quad k = 1, \dots, 16.$$

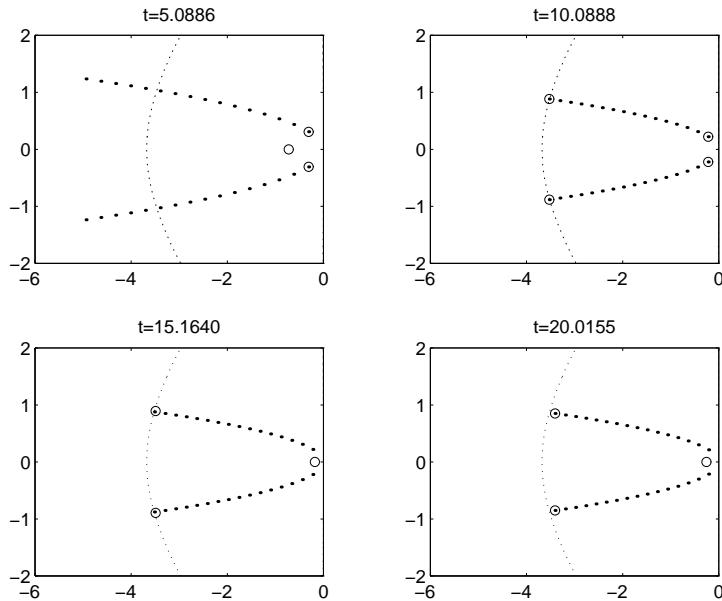


Figure 4: Exact (dots) and approximate (circles) eigenvalues for problem with complex eigenvalues at four different times. The dotted line shows the boundary of the stability region.

In the four snapshots of Figure 4, we show the location of the eigenvalues of hA and the eigenvalues obtained with the new stiffness detection scheme. The dots show the eigenvalues of hA , the circles are the approximating eigenvalues resulting from the stiffness detection scheme, while the dotted line show the stability region of the method used, RKF4(5).

5 Conclusion

In this paper we have proposed a new way of detecting stiffness when integrating initial value problems with an explicit Runge-Kutta method. The approach was based on Arnoldi's method for projecting an eigenvalue problem onto a corresponding Krylov subspace. We made the observation that the stages of an explicit Runge-Kutta method locally approximates a basis of a Krylov subspace of the Jacobian of the initial value problem. We thereby obtained an inexpensive approximation of the dominant eigenvalues of the Jacobian. In this paper, we have elected to use this information for the purpose of detecting stiffness, but there may be other applications as well. For instance, it may be possible to use the information on the dominant part of the eigenspace of the Jacobian to filter out stiffness, thereby making explicit methods feasible for integration of mildly stiff problems. This application was not pursued here and it is probably only feasible in the case that there are only a few dominant eigenvalues. There are other stiffness detection schemes, like the comparison formula pair that may seem simpler and less computationally expensive than the one presented here. It should be pointed out that the extra cost of the new technique is rewarded through the access to more detailed information about the problem. It is for instance well known that problems with eigenvalues on or near the imaginary axis are usually best solved with special

methods like for instance symplectic Runge-Kutta methods. The technique presented here can give such information as opposed to for instance the comparison formula technique. The power method of Shampine [12] also yields an approximation to the dominant eigenvalue, and he mentions briefly how to obtain approximations to the two most dominant eigenvalues. Our approach here may give somewhat more information since additional eigenvalues can be approximated. One should however note that when the problem has been stiff for some time, all the stages will tend to become rich in the direction of the most dominant eigenvector, hence, there is little information left to recover other pairs of eigenvectors/eigenvalues, and attempts to do so will typically give poor results, due to rounding errors.

There are problems where the new technique fails to give good estimates to the eigenvalues of the Jacobian. A good example is the class of problems with extremely non-normal Jacobians. For such cases, the problem of finding eigenvalues is itself ill-posed. At best, one may expect to find approximations to the pseudospectrum of the matrix, see Toh and Trefethen [13]. In a paper by Higham and Trefethen [6], it is shown that it may be more useful to consider the pseudospectrum than the spectrum of the Jacobian when assessing the stiffness of a problem. Our experience is that for highly non-normal problems it seems hard to extract useful information about the stiffness from the information obtained by the stages of an explicit Runge-Kutta method.

Finally, the computational cost of the new stiffness detection scheme is proportional to the dimension N of the ODE system, and the coefficient of the N -term is approximately s^2 , where s is the number of stages used. No additional function evaluations are needed. The cost of the comparison formula pair will be approximately $s \cdot N$. Especially for large problems, and for problems where high quality information on the spectrum of the Jacobian is needed, we believe that the stiffness detection scheme presented here is at least a worth while alternative to existing techniques.

Acknowledgement. We would like to thank Professor Ken R. Jackson at the University of Toronto for suggesting some of the ideas used in this paper. We would also like to thank Des Higham, at the University of Strathclyde for useful comments on an earlier version of the manuscript.

References

- [1] C.F. Curtiss and J.O. Hirschfelder, *Integration of stiff equations*, Proc. of the National Academy of Sciences of U.S., 38 (1952), pp 235–243.
- [2] K. Dekker and J.G. Verwer, *Stability of Runge-Kutta methods for stiff nonlinear differential equations*, North-Holland, 1984.
- [3] K. Ekeland and E. Øines, *Detecting Stiffness in Explicit Runge-Kutta Methods*, term project report, Dept. of Math. Sci., NTNU, Norway, 1996.
- [4] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Oxford, 1983.
- [5] E. Hairer and G. Wanner, *Solving Ordinary Differential Equations II, 2nd ed.*, Springer, 1996.

- [6] D.J. Higham and L.N. Trefethen, *Stiffness of ODE's*, BIT 33 (1993), pp 285–303.
- [7] J.D. Lambert, *Numerical Methods for Ordinary Differential Systems*, Wiley, 1991.
- [8] B.C. Robertson, *Detecting Stiffness With Explicit Runge-Kutta Methods*, Technical Report 193/87, Department of Computer Science, University of Toronto, 1987.
- [9] Y. Saad, *Numerical Methods For Large Eigenvalue Problems*, Manchester University Press, UK, 1992.
- [10] L.F. Shampine, *Stiffness and nonstiff differential equations solvers, II: Detecting Stiffness with Runge-Kutta methods*, ACM Transactions on Mathematical Software, 3 (1977), pp. 44–53.
- [11] L.F. Shampine, *Lipschitz constants and robust ODE codes*, in Computational Methods in Nonlinear Mechanics, J. Oden, ed., North-Holland, Amsterdam, 1980, pp 427–449.
- [12] L.F. Shampine, *Diagnosing Stiffness for Runge-Kutta Methods*, SIAM J. Sci. Stat. Comput., (1991), pp 260–272.
- [13] K.C. Toh and L.N. Trefethen, *Calculation of pseudospectra by the Arnoldi iteration*, SIAM J. Sci. Comput. 17 (1996), pp 1–15.