Noname manuscript No.
(will be inserted by the editor)

A Linearly Fourth Order Multirate Runge-Kutta Method with Error Control

Pak-Wing Fok

Received: date / Accepted: date

Abstract To integrate large systems of locally coupled ordinary differential equations (ODEs) with disparate timescales, we present a multirate method with error control that is based on the Cash-Karp Runge-Kutta (RK) formula. The order of multirate methods often depends on interpolating certain solution components with a polynomial of sufficiently high degree. By using cubic interpolants and analyzing the method applied to a simple test equation, we show that our method is fourth order linearly accurate overall. Furthermore, the size of the region of absolute stability is increased when taking many "micro-steps" within a "macro-step." Finally, we demonstrate our method on three simple test problems to confirm fourth order convergence.

Keywords Multirate · Runge-Kutta · Interpolation

1 Introduction

In this paper, we present a fourth order linearly accurate multirate method with error control, that is suited for integrating large systems of locally coupled ordinary differential equations (ODEs) whose solutions exhibit different time scales. The method is based on a Cash-Karp Runge-Kutta (RK) formula [16] with "latent" and "active" components coupled together through a third order interpolant. The latent components are assumed to be non-stiff while the active components may or may not be stiff.

A multirate method is one that can take different step sizes for different components of the solution [8]. When might such a need for different time steps arise? One situation where multirate methods may be more efficient than single rate ones is when a few of the components contain time singularities or move on short time scales. In this case, explicit single rate methods may use a small global time step for all the components, whereas a multirate one could employ small time steps just for singular/fast ones, therefore improving the speed and efficiency of solution.

P.-W. Fok
412 Ewing Hall, University of Delaware, Newark, DE 19716, USA
E-mail: pakwing@udel.edu

Multirate methods typically partition the solution into "active" and "latent" components. The active (latent) components are integrated using "micro" ("macro") steps and the integrators used for the micro and macro steps do not have to be the same; however they must be coupled together. How this coupling can be achieved without reducing the method order is the topic of most investigations.

Early multirate schemes were studied by Andrus [1], and Gear and Wells [8]. Kvaerno and other authors [9,3,5] advocate the use of Multirate Partitioned Runge-Kutta formulas (MPRK). The distinguishing feature of this class of methods is that macro- and micro-integrators are coupled through the intermediate stage function evaluations. In [12], a multirate θ-method is studied; different values of θ give rise to forward Euler, backward Euler and Trapezoidal integrators and different orders of interpolation were tried. It was found that poorly chosen interpolants could give rise to order reduction or solution instability. Multirate methods are often based on existing integrators. Multirate RK has already been discussed above. However, equipped with suitable interpolants, researchers have successfully implemented multirate versions of Rosenbrock [18,17], Adams [6] and BDF [20] methods. Further treatments can be found in [2,13–15,21], and the references therein.

The overall order of a multirate method usually depends on the accuracy of interpolation. Multirate methods have generally been restricted to low order due to difficulties in constructing suitable interpolants. However, in [17], a fourth order Rosenbrock method was developed and in [4], extrapolated multirate methods were studied. The advantage of the methods in [4] is that they are easily parallelized. While the base methods are low order (either explicit or implicit Euler along with zeroth order interpolation), they can be used to find very accurate numerical solutions via Richardson extrapolation.

Our method shares some similarities with the MPRK methods [9,3] in the sense that RK methods are the base schemes used to advance the solution. However, our method is 4th order whereas the authors in [9,3] investigate 2nd and 3rd order (embedded) methods. Furthermore, we use interpolation to couple the micro- and macro-integrators. Because we can generate dense output for the latent components, we can implement error control and adaptive time stepping in the micro-integrator; the number of micro-steps within a macro-step is free to vary between macro-steps. Because we use a "slowest first" strategy, adaptive time stepping in the macro-integrator is independent of the micro-integrator and can be implemented without additional complications. Our method also shares similarities with [18] in that active components are detected by examining the error of each component. However, in our method, automatic step size selection is simple to implement and follows standard protocols [16].

In section 2 we give the details of the numerical method. In section 3, we investigate the order and absolute stability of our method. We discuss the important issue of interpolation in section 4 and validate our code and present our results in section 5. Finally, we summarize our findings with a conclusion in section 6.

2 Algorithm

The goal of our method is to efficiently solve large systems of ordinary differential equations whose solutions evolve on very different time scales. In this paper, we

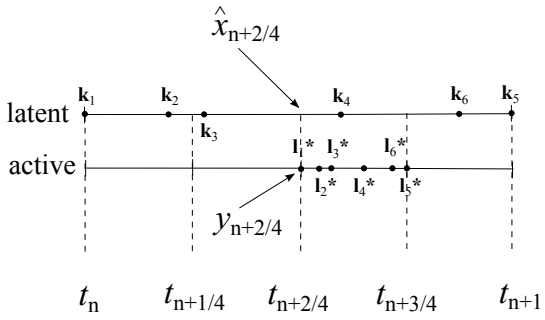

Fig. 1 Intermediate stage function evaluations and timeslabs in a Cash-Karp RK multirate scheme with $m = 4$ micro-steps and $t_{n+1} - t_n = h$. The $\mathbf{k}_j$ and $\mathbf{l}_j^*$ denote intermediate stage function evaluations within a macro-step and micro-step respectively. Symbols with hat accents are calcuated through interpolation.

consider systems that can be represented as

$$\dot{\mathbf{X}} = \mathbf{f}_0(t, \mathbf{X}, \mathbf{x}), \tag{1}$$

$$\dot{\mathbf{x}} = \mathbf{f}_1(t, \mathbf{X}, \mathbf{x}, \mathbf{y}), \tag{2}$$

$$\dot{\mathbf{y}} = \mathbf{f}_2(t, \mathbf{x}, \mathbf{y}), \tag{3}$$

where for integers N_0, N_1, N_2 and $N = N_0 + N_1 + N_2$, the vectors $\mathbf{X} \in \mathbb{R}^{N_0}$, $\mathbf{x} \in \mathbb{R}^{N_1}$, $\mathbf{y} \in \mathbb{R}^{N_2}$; $\mathbf{f}_0 : (\mathbb{R}^+ \times \mathbb{R}^{N_0} \times \mathbb{R}^{N_1}) \mapsto \mathbb{R}^{N_0}$, $\mathbf{f}_1 : (\mathbb{R}^+ \times \mathbb{R}^{N_0} \times \mathbb{R}^{N_1} \times \mathbb{R}^{N_2}) \mapsto \mathbb{R}^{N_1}$ and $\mathbf{f}_2 : (\mathbb{R}^+ \times \mathbb{R}^{N_1} \times \mathbb{R}^{N_2}) \mapsto \mathbb{R}^{N_2}$. Furthermore, we assume that $N_0 \gg N_1, N_2$. Following [9,3], we have broadly grouped our solution components into "latent" ($\mathbf{X}$, $\mathbf{x}$) and "active" ($\mathbf{y}$) components. The latent components $\mathbf{X}$ (which are the majority of the latent components) and the active components $\mathbf{y}$ are decoupled: $\dot{\mathbf{X}}$ does not depend on $\mathbf{y}$ and $\dot{\mathbf{y}}$ does not depend on $\mathbf{X}$. However, the evolutions of $\mathbf{X}$ and $\mathbf{y}$ are not independent in the sense that they are indirectly coupled with each other through a small subset of latent components $\mathbf{x}$.

Systems such as (1)-(3) actually arise in many applications. For example, when partial differential equations are discretized using the method of lines, the result is usually a large system of *locally coupled* ordinary differential equations. In this case, a small number of active solution components, $\mathbf{y}$, may change on a fast time scale while the remaining latent ones, $\mathbf{X} \cup \mathbf{x}$, change relatively slowly. Furthermore, because the system is locally coupled, the active components only influence (through their rate functions) a small subset of latent components, $\mathbf{x}$. For our method to be competitive, we stress that the ODE system must be locally coupled (i.e. the stencil of dependence in the rate function must be localized) and the solution must contain only a small number of active components at any given time.

Our algorithm advances the latent components $(\mathbf{X}, \mathbf{x})$ using a single *macro-step* of size h. It advances the active components $\mathbf{y}$ using the same time stepper with *micro-steps* of smaller size. The partioning into latent and active components will, in general, be time-dependent, *i.e.* the set of latent and active components can change from macro-step to macro-step. The description of the algorithm below and analysis assumes a fixed micro-step of size h/m where m is an integer. Because both latent and active components are advanced using embedded RK,

the extension to implement error control and adaptive time stepping, based on estimation of the local truncation error, is straightforward [16]. Our multirate RK algorithm to advance one macro-step is described below. The coefficients a_i, b_{ij}, c_j and $\tilde{c}_j$ are the standard Cash-Karp RK coefficients; see Table 1.

1. For an integer n, fix a time point t_n and compute the intermediate stage function evaluations for latent and active components over a macro-step of size h:

$$\mathbf{K}_i = h\mathbf{f}_0\left(t_n + c_i h,\ \mathbf{X}_n + \sum_{j=1}^{i-1} a_{ij}\mathbf{K}_j\ ,\ \mathbf{x}_n + \sum_{j=1}^{i-1} a_{ij}\mathbf{k}_j\right), \tag{4}$$

$$\mathbf{k}_i = h\mathbf{f}_1\left(t_n + c_i h,\ \mathbf{X}_n + \sum_{j=1}^{i-1} a_{ij}\mathbf{K}_j\ ,\ \mathbf{x}_n + \sum_{j=1}^{i-1} a_{ij}\mathbf{k}_j,\ \mathbf{y}_n + \sum_{j=1}^{i-1} a_{ij}\mathbf{l}_j\right), \tag{5}$$

$$\mathbf{l}_i = h\mathbf{f}_2\left(t_n + c_i h,\ \mathbf{x}_n + \sum_{j=1}^{i-1} a_{ij}\mathbf{k}_j\ ,\ \mathbf{y}_n + \sum_{j=1}^{i-1} a_{ij}\mathbf{l}_j\right), \tag{6}$$

   for $i = 1, \ldots, 6$. In (4)-(6), $\mathbf{K}_i \in \mathbb{R}^{N_0}$, $\mathbf{k}_i \in \mathbb{R}^{N_1}$ and $\mathbf{l}_i \in \mathbb{R}^{N_2}$.
2. Advance all latent components:

$$\mathbf{X}_{n+1} = \mathbf{X}_n + \sum_{i=1}^{6} b_i\mathbf{K}_i \qquad \text{(fourth order)}, \tag{7}$$

$$\mathbf{x}_{n+1} = \mathbf{x}_n + \sum_{i=1}^{6} b_i\mathbf{k}_i \qquad \text{(fourth order)}, \tag{8}$$

$$\tilde{\mathbf{X}}_{n+1} = \mathbf{X}_n + \sum_{i=1}^{6} \tilde{b}_i\mathbf{K}_i \qquad \text{(fifth order)}, \tag{9}$$

$$\tilde{\mathbf{x}}_{n+1} = \mathbf{x}_n + \sum_{i=1}^{6} \tilde{b}_i\mathbf{k}_i \qquad \text{(fifth order)}. \tag{10}$$

   Our method does not use local extrapolation, so eqs. (7) and (8) advance the solution while (9) and (10) are used to calculate the corresponding error.
3. For $\ell \in \{0, 1, \ldots, m-1\}$, advance the active components through

$$\mathbf{y}_{n+(\ell+1)/m} = \mathbf{y}_{n+\ell/m} + \sum_{d=1}^{6} b_d\mathbf{l}_d^*, \tag{11}$$

$$\tilde{\mathbf{y}}_{n+(\ell+1)/m} = \mathbf{y}_{n+\ell/m} + \sum_{d=1}^{6} \tilde{b}_d\mathbf{l}_d^*. \tag{12}$$

4. The intermediate stage function evaluations within a micro-step are computed through

$$\mathbf{l}_d^* = \frac{h}{m}\mathbf{f}_2\left(t_{n+\frac{\ell}{m}} + \frac{c_d h}{m}\ ,\ \hat{\mathbf{x}}_{n+\frac{\ell+c_d}{m}}\ ,\ \mathbf{y}_{n+\frac{\ell}{m}} + \sum_{j=1}^{d-1} a_{dj}\mathbf{l}_j^*\right), \tag{13}$$

for $d = 1, \ldots, 6$. In (13), we interpolate only the latent components that are coupled to active components via the formula

$$\hat{\mathbf{x}}_{n+\chi} = \mathbf{x}_n + \mathbf{k}_1\chi + \frac{\chi^2}{2}\left(-\frac{8\mathbf{k}_1}{3} + \frac{25\mathbf{k}_4}{6} - \frac{3\mathbf{k}_5}{2}\right) + \frac{\chi^3}{6}\left(\frac{10\mathbf{k}_1}{3} - \frac{25\mathbf{k}_4}{3} + 5\mathbf{k}_5\right), \tag{14}$$

for any $0 < \chi < 1$. We will derive (14) in section 4.

i	c_i	a_{ij}						b_i	$\tilde{b}_i$
1	0							$\frac{2825}{27648}$	$\frac{37}{378}$
2	$\frac{1}{5}$	$\frac{1}{5}$						0	0
3	$\frac{3}{10}$	$\frac{3}{40}$	$\frac{9}{40}$					$\frac{18575}{48384}$	$\frac{250}{621}$
4	$\frac{3}{5}$	$\frac{3}{10}$	$-\frac{9}{10}$	$\frac{6}{5}$				$\frac{13525}{55296}$	$\frac{125}{594}$
5	1	$-\frac{11}{54}$	$\frac{5}{2}$	$-\frac{70}{27}$	$\frac{35}{27}$			$\frac{277}{14336}$	0
6	$\frac{7}{8}$	$\frac{1631}{55296}$	$\frac{175}{512}$	$\frac{575}{13824}$	$\frac{44275}{110592}$	$\frac{253}{4096}$		$\frac{1}{4}$	$\frac{512}{1771}$
	$j =$	1	2	3	4	5	6		

Table 1 Table of coefficients for Cash-Karp embedded Runge-Kutta formula. The coefficients b_i and $\tilde{b}_i$ are used for fourth and fifth order formula respectively. See [16] for details.

Figure 1 summarizes the algorithm and shows the macro timesteps (indexed by n in our algorithm) along with their intermediate stage function evaluations (indexed by i), and $m = 4$ micro timesteps (indexed by ℓ) along with their intermediate stage function evaluations (indexed by d). For a single rate scheme with no interpolation, the integrators (11) and (12) are fourth and fifth order respectively. Because the multirate scheme interpolates using (14), eq. (12) actually reduces to fourth order. However, the error can still be estimated from $||\mathbf{y}_{n+(\ell+1)/m} - \tilde{\mathbf{y}}_{n+(\ell+1)/m}||$. Because we are specializing to locally coupled systems, $\mathbf{l}_d^*$, $d = 1, \ldots, 6$, in eq. (13) only has to be computed for N_2 components and interpolation (14) only has to be done for N_1 components (recall that both $N_1, N_2 \ll N_0$).

We now discuss some technical details of our method.

Dynamic Partioning: The partitioning into latent and active components is done by estimating the local truncation error (LTE) for each component of the solution. Therefore the solution components labeled as $\mathbf{X}$, $\mathbf{x}$ and $\mathbf{y}$ can change from macro-step to macro-step. Components whose errors are larger than $\delta \times$ (maximum error) (where $\delta \ll 1$) are flagged as active and are corrected through a second integration using the micro-integrator (see below). In the numerical examples discussed later on, δ can range from 10^{-12} to 10^{-2}. When $\delta = 1$, there are no active components and our method effectively reduces to a single rate method. The remaining unflagged components are classified as latent. The `flag` vector – consisting of ones (active) and zeros (latent) – is then searched to find boundary

indices $[r_j, s_j]$ for $j = 1, 2, \ldots, \nu$ which define ν zones of activity. Zones of activity may consist of just a single solution component j in which case $r_j = s_j$.

For coupled ODEs with a local stencil of size q, an odd integer, zones of activity must be separated by at least $(q-1)/2$ latent components. If active zones are separated by fewer than $(q-1)/2$ components, they are combined into a single active zone and latent components within the zone are integrated along with the other active components using the micro-integrator. In a pentadiagonal case with $q = 5$, we have $\dot{u}_j = F_j(u_{j-2}, u_{j-1}, u_j, u_{j+1}, u_{j+2})$ for some rate function F_j and a single active component u_j in the bulk can be integrated a second time at relatively little cost if the two latent components on either side $u_{j\pm1}$, $u_{j\pm2}$ are interpolated.

In section 5, we study examples with a single zone of activity, although our methods easily generalize to multiple zones. In this case, for a given integer P, we may expand the zone of activity from $[r, s]$ to $[r - P, s + P]$ to be conservative. (This strategy ensures that borderline active/latent components are always classified as active.) The integer P must also be chosen so that indices of interpolated components lie between 1 and N. For example, with two interpolated components on either size of the active zone, upon determining $[r, s]$ we conservatively update with $r \leftarrow \max(r - P, 3)$ and $s \leftarrow \min(s + P, N - 2)$. In the numerical examples of section 5, we take $P = 10$.

Macro-Integration: After taking a trial macro-step and flagging the active components, the algorithm checks if the unflagged errors are smaller than a preset tolerance. If they are, the trial step is successful and the macro-step size is scaled up according to the largest unflagged error. If there are latent components which have error larger than the preset tolerance, the trial step is unsuccessful and has to be retaken. In this case the macro-step size is scaled down according to the largest unflagged error.

Micro-Integration: Although the method is described above for m micro-steps each of size h/m, in practice we implement (11)-(13) with error control/adaptive time stepping. The number of micro-steps in each macro-step can vary from macro-step to macro-step and the micro-step size can also vary within a macro-step. The micro-step size is chosen so that the accumulated errors from the m micro-steps is less than a preset tolerance and the error from one macro-step is also controlled to be less than this tolerance. Whether a single macro-step is taken for a latent component, or m micro-steps are taken for an active component, the error committed is approximately the same.

A flowchart summary of our method is given in Figure 2.

3 Consistency and Order

Analysis of Single Rate Method. We first analyze a single-rate method to introduce notation that will feature in the multirate analysis. This section also reuses many of the symbols in the scheme (4)-(14).

Suppose we apply a single rate Cash-Karp scheme to the system of ODEs

$$\dot{\mathbf{u}} + M\mathbf{u} = 0, \qquad M = \begin{pmatrix} M_{00} & M_{01} & M_{02} \\ M_{10} & M_{11} & M_{12} \\ M_{20} & M_{21} & M_{22} \end{pmatrix}, \tag{15}$$

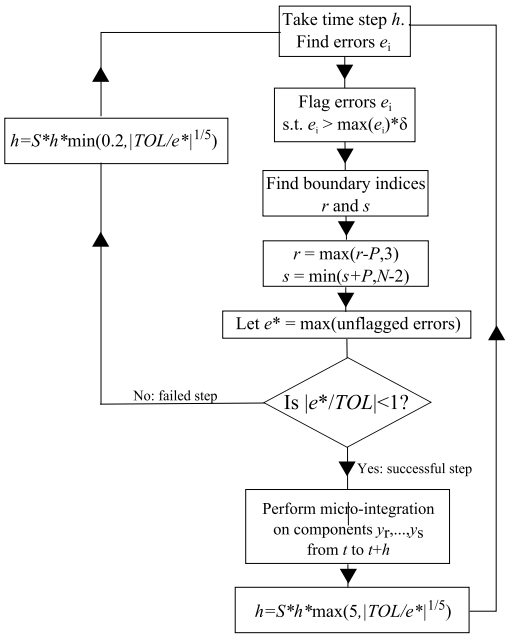

Fig. 2 Flowchart description of adaptive-time stepping multirate method for the ODE system $\dot{y} = f(t, y)$. The time step tolerance is TOL and $\delta \ll 1$ and $0 < S < 1$ are required constants. The maximum stepsize amplification and contraction factors of 5 and 0.2 are arbitrary and ensure that h does not change too quickly as the agorithm runs. The indices r and s define the boundary components of the active region.

where $\mathbf{u} = (\mathbf{X}^T, \mathbf{x}^T, \mathbf{y}^T)^T$, $\mathbf{X} \in \mathbb{R}^{N_0}$, $\mathbf{x} \in \mathbb{R}^{N_1}$, $\mathbf{y} \in \mathbb{R}^{N_2}$, and the submatrices $M_{ij} \in \mathbb{R}^{N_i \times N_j}$. In a single rate method, all components advance using a macro-step of size h. Then the intermediate stage function evaluations satisfy

$$(I_{6N} + hA \otimes M) \begin{bmatrix} \mathbf{K} \\ \mathbf{k} \\ \mathbf{l} \end{bmatrix} = -h\mathbf{1} \otimes M \begin{bmatrix} \mathbf{X}_n \\ \mathbf{x}_n \\ \mathbf{y}_n \end{bmatrix}, \tag{16}$$

where $\mathbf{1} = (1, 1, 1, 1, 1, 1)^T$, the entries of the matrix A are the a_{ij} in Table 1 and I_n is the $n \times n$ identity matrix. Solving for $\mathbf{K}$, $\mathbf{k}$ and $\mathbf{l}$, we find that

$$L_0\mathbf{K}(h) = U_{00}\mathbf{X}_n + U_{01}\mathbf{x}_n + U_{02}\mathbf{y}_n, \tag{17}$$
$$L_1\mathbf{k}(h) = U_{10}\mathbf{X}_n + U_{11}\mathbf{x}_n + U_{12}\mathbf{y}_n, \tag{18}$$
$$L_2\mathbf{l}(h) = U_{20}\mathbf{X}_n + U_{21}\mathbf{x}_n + U_{22}\mathbf{y}_n, \tag{19}$$

where the L_i and U_{ij} are lengthy (but known) functions of h that are defined in the appendix. Then the fourth order scheme in Cash-Karp Runge Kutta updates through

$$\mathbf{X}_{n+1} = \mathbf{X}_n + (\mathbf{b}^T \otimes I_{N_0})L_0^{-1}(h)\left[U_{00}(h)\mathbf{X}_n + U_{01}(h)\mathbf{x}_n + U_{02}(h)\mathbf{y}_n\right], \tag{20}$$
$$\mathbf{x}_{n+1} = \mathbf{x}_n + (\mathbf{b}^T \otimes I_{N_1})L_1^{-1}(h)\left[U_{10}(h)\mathbf{X}_n + U_{11}(h)\mathbf{x}_n + U_{12}(h)\mathbf{y}_n\right], \tag{21}$$
$$\mathbf{y}_{n+1} = \mathbf{y}_n + (\mathbf{b}^T \otimes I_{N_2})L_2^{-1}(h)\left[U_{20}(h)\mathbf{X}_n + U_{21}(h)\mathbf{x}_n + U_{22}(h)\mathbf{y}_n\right], \tag{22}$$

while the fifth order scheme updates through a similar scheme to (20)-(22) but with $\mathbf{b}$ replaced with $\tilde{\mathbf{b}}$. We see that although $\mathbf{X}$ and $\mathbf{y}$ are not directly coupled together in eqs. (1) and (3), in one step of single rate Cash-Karp Runge Kutta, $\mathbf{y}_n$ affects $\mathbf{X}_{n+1}$ through (20) and $\mathbf{X}_n$ affects $\mathbf{y}_{n+1}$ through (22).

Multirate Analysis. We first study the fourth order multirate method given by eqs. (7), (8) and (11). In a multirate scheme, eqs. (20) and (21) remain identical but (22) is replaced by a scheme that takes m micro-steps of size h/m. The goal in this section is to derive the multirate equivalent of (22). Using eq. (13), the intermediate stage function evaluations for the active components satisfy

$$\mathbf{l}_d^* = -\frac{hM_{21}}{m}\hat{\mathbf{x}}_{n+\frac{\ell+c_d}{m}} - \frac{hM_{22}}{m}\left(\mathbf{y}_{n+\frac{\ell}{m}} + \sum_{j=1}^{d-1} a_{dj}\mathbf{l}_j^*\right), \tag{23}$$

for $d = 1, \ldots, 6$, where

$$\hat{\mathbf{x}}_{n+\frac{\ell+\mathbf{c}}{m}} \equiv \begin{pmatrix} \hat{\mathbf{x}}_{n+(\ell+c_1)/m} \\ \hat{\mathbf{x}}_{n+(\ell+c_2)/m} \\ \vdots \\ \hat{\mathbf{x}}_{n+(\ell+c_6)/m} \end{pmatrix} \in \mathbb{R}^{6N_1}, \tag{24}$$

is a vector of interpolated values for $\mathbf{x}$ from $t_{n+\ell/m}$ to $t_{n+(\ell+1)/m}$, and $c_1, \ldots, c_6$ are taken from Table 1. Taking $\mathbf{l}^* = (\mathbf{l}_1^{*T}, \mathbf{l}_2^{*T}, \ldots, \mathbf{l}_6^{*T})^T$, (23) is more compactly written as

$$\left[I_{6N_2} + \frac{hA}{m} \otimes M_{22}\right]\mathbf{l}^* = -\frac{h}{m}(I_6 \otimes M_{21})\hat{\mathbf{x}}_{n+\frac{\ell+\mathbf{c}}{m}} - \frac{h}{m}(\mathbf{1} \otimes M_{22})\mathbf{y}_{n+\frac{\ell}{m}}. \tag{25}$$

From the definition of the interpolant (14), we can use kronecker products to write the interpolated values in (24) as

$$\hat{\mathbf{x}}_{n+(\ell+\mathbf{c})/m} = \mathbf{1} \otimes \mathbf{x}_n + (\Omega(\ell) \otimes I_{N_1})(Q \otimes I_{N_1})\mathbf{k}, \tag{26}$$

where

$$Q = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ -\frac{8}{3} & 0 & 0 & \frac{25}{6} & -\frac{3}{2} & 0 \\ \frac{10}{3} & 0 & 0 & -\frac{25}{3} & 5 & 0 \end{bmatrix}, \qquad [\Omega(\ell)]_{ij} = \frac{1}{j!}\left(\frac{\ell + c_i}{m}\right)^j, \tag{27}$$

for $i = 1, \ldots, 6, j = 1, 2, 3$ and $\ell = 0, \ldots, m-1$.

We now analyze the micro timestepper. From eq. (11),

$$\mathbf{y}_{n+(\ell+1)/m} = \mathbf{y}_{n+\ell/m} + (\mathbf{b}^T \otimes I_{N_2})\mathbf{l}^*, \tag{28}$$

$$= \zeta_1 \hat{\mathbf{x}}_{n+(\ell+\mathbf{c})/m} + \zeta_2 \mathbf{y}_{n+\ell/m}, \tag{29}$$

using eq. (25) and the matrices $\zeta_1 \in \mathbb{R}^{N_2 \times 6N_1}$ and $\zeta_2 \in \mathbb{R}^{N_2 \times N_2}$ are given by

$$\zeta_1 = -\frac{h}{m}(\mathbf{b}^T \otimes I_{N_2})\left[I_{6N_2} + \frac{hA}{m} \otimes M_{22}\right]^{-1}(I_6 \otimes M_{21}), \tag{30}$$

$$\zeta_2 = I_{N_2} - \frac{h}{m}(\mathbf{b}^T \otimes I_{N_2})\left[I_{6N_2} + \frac{hA}{m} \otimes M_{22}\right]^{-1}(\mathbf{1} \otimes M_{22}). \tag{31}$$

Eq. (29) gives the active solution at time $t_{n+(\ell+1)/m}$ in terms of the the active solution at time $t_{n+\ell/m}$ and interpolated values. Implementing the micro-integration, we iterate (29) m times and accumulate the interpolated values to obtain

$$\mathbf{y}_{n+1} = \sum_{j=0}^{m-1} \zeta_2^{m-j-1}\zeta_1 \hat{\mathbf{x}}_{n+(j+\mathbf{c})/m} + \zeta_2^m \mathbf{y}_n. \tag{32}$$

Finally, we substitute for $\mathbf{k}$ in (26) using (18) and then substitute the resulting expression for $\hat{\mathbf{x}}_{n+(\ell+\mathbf{c})/m}$ into (32).

In summary, when we apply the multirate scheme to the linear system (15), the solution at times t_{n+1} and t_n are related through

$$\mathbf{u}_{n+1} = S(hM)\mathbf{u}_n, \tag{33}$$

where $S = \begin{bmatrix} S_{00} & S_{01} & S_{02} \\ S_{10} & S_{11} & S_{12} \\ S_{20} & S_{21} & S_{22} \end{bmatrix}$ and the submatrices $S_{ij} \in \mathbb{R}^{N_i \times N_j}$ are defined as

$$S_{ij} = I_{N_i}\delta_{ij} + (\mathbf{b}^T \otimes I_{N_i})L_i^{-1}U_{ij}, \qquad i \in \{0,1\}, j \in \{0,1,2\}, \tag{34}$$

$$S_{20} = \sum_{j=0}^{m-1} \zeta_2^{m-j-1}\zeta_1(\Omega(j) \otimes I_{N_1})(Q \otimes I_{N_1})L_1^{-1}U_{10}, \tag{35}$$

$$\begin{aligned} S_{21} = &\sum_{j=0}^{m-1} \zeta_2^{m-j-1}\zeta_1(\mathbf{1} \otimes I_{N_1}) \\ &+ \sum_{j=0}^{m-1} \zeta_2^{m-j-1}\zeta_1(\Omega(j) \otimes I_{N_1})(Q \otimes I_{N_1})L_1^{-1}U_{11}, \end{aligned} \tag{36}$$

$$S_{22} = \zeta_2^m + \sum_{j=0}^{m-1} \zeta_2^{m-j-1}\zeta_1(\Omega(j) \otimes I_{N_1})(Q \otimes I_{N_1})L_1^{-1}U_{12}, \tag{37}$$

where δ_{ij} is the (scalar) kronecker delta. Note that while eqs. (20)-(22) depend on U_{ij}, $i = 1,2,3$, $j = 1,2,3$, eqs. (34)-(37) are independent of U_{20}, U_{21} and U_{22}.

Applying the same analysis to eqs. (9), (10) and (12), we have

$$\tilde{\mathbf{u}}_{n+1} = \tilde{S}(hM)\mathbf{u}_n, \tag{38}$$

where $\tilde{S} = \begin{bmatrix} \tilde{S}_{00} & \tilde{S}_{01} & \tilde{S}_{02} \\ \tilde{S}_{10} & \tilde{S}_{11} & \tilde{S}_{12} \\ \tilde{S}_{20} & \tilde{S}_{21} & \tilde{S}_{22} \end{bmatrix}$ and the submatrices $\tilde{S}_{ij}$ have similar definitions to (34)-(37) but with $\mathbf{b}$ replaced by $\tilde{\mathbf{b}}$ and ζ_1 and and ζ_2 replaced by $\tilde{\zeta}_1$ and $\tilde{\zeta}_2$. Matrices $\tilde{\zeta}_1$ and $\tilde{\zeta}_2$ have similar definitions to (30) and (31) but with $\mathbf{b}$ replaced by $\tilde{\mathbf{b}}$.

By expanding S and $\tilde{S}$ in Taylor series for small h, one can show that

$$S(hM) = I_N - hM + \frac{(hM)^2}{2} - \frac{(hM)^3}{6} + \frac{(hM)^4}{24} + Eh^5 + O(h^6), \tag{39}$$

$$\tilde{S}(hM) = I_N - hM + \frac{(hM)^2}{2} - \frac{(hM)^3}{6} + \frac{(hM)^4}{24} + \tilde{E}h^5 + O(h^6), \tag{40}$$

for different matrices $E \in \mathbb{R}^{N\times N}$ and $\tilde{E} \in \mathbb{R}^{N\times N}$. Since $(-M)^j \mathbf{u} = \mathbf{u}^{(j)}(t)$ for $j \geq 0$, eqs. (33) and (38) along with eqs. (39) and (40) imply

$$\mathbf{u}_{n+1} = \mathbf{u}(t_n) + h\mathbf{u}'(t_n) + \frac{h^2\mathbf{u}''(t_n)}{2} + \frac{h^3\mathbf{u}'''(t_n)}{6} + \frac{h^4\mathbf{u}^{(4)}(t_n)}{24} + O(h^5), \quad (41)$$

$$\tilde{\mathbf{u}}_{n+1} = \mathbf{u}(t_n) + h\mathbf{u}'(t_n) + \frac{h^2\mathbf{u}''(t_n)}{2} + \frac{h^3\mathbf{u}'''(t_n)}{6} + \frac{h^4\mathbf{u}^{(4)}(t_n)}{24} + O(h^5), \quad (42)$$

with *different* implied constants in the $O(h^5)$ terms.

Single rate Cash-Karp Runge-Kutta contains both fourth and fifth order integrators. However, using the interpolant (14), we see from (41) and (42) that in the multirate scheme, both integrators are linearly 4th order accurate. Specifically, the compound step defined by applying (12) m times is linearly *fourth order*, although the constants $\{\tilde{b}_1, \ldots, \tilde{b}_6\}$ are normally associated with a fifth order formula. Nevertheless, we can still estimate the micro-step error by $||\mathbf{y}_{n+(i+1)/m} - \tilde{\mathbf{y}}_{n+(i+1)/m}||_\infty$. This is not too different than step doubling methods where the numerical solution after two half-steps and a single whole-step also have the same order of error. Our numerical results in Section 5 show that, in practice, this way of estimating the error does not reduce the overall accuracy of the method.

In summary, we have shown that the method of taking one macro-step of size h in components $\mathbf{X}$ and $\mathbf{x}$, and taking m micro-steps of size h/m in components $\mathbf{y}$ is linearly fourth order overall: the associated error is $O(h^5)$ when applied to the simple test equation (15). Because we interpolate latent components $\mathbf{x}$ using a cubic polynomial, the fifth order formula for the micro timestepper reduces to fourth order.

Absolute Stability of Multirate Steps. To analyze absolute stability of our multirate method, we study the matrix S in eq. (33). To simplify the analysis, we take $N_0 = N_1 = N_2 = 1$ and $M_{00} = M_{01} = M_{10} = M_{02} = M_{20} = 0$, so that

$$M = \begin{pmatrix} 0 & 0 & 0 \\ 0 & M_{11} & M_{12} \\ 0 & M_{21} & M_{22} \end{pmatrix}, \quad (43)$$

where M_{ij}, $i, j \in \{1, 2\}$ are scalars. Note that $X(t)$ is constant in time and decoupled from $x(t)$ and $y(t)$. Therefore we only need to analyze a 2×2 system of ODEs, and our multirate scheme reduces to

$$\begin{pmatrix} x_{n+1} \\ y_{n+1} \end{pmatrix} = \underbrace{\begin{pmatrix} S_{11} & S_{12} \\ S_{21} & S_{22} \end{pmatrix}}_{S_R} \begin{pmatrix} x_n \\ y_n \end{pmatrix}. \quad (44)$$

Given the macro-step size to micro-step size ratio m, the requirement that the solution at time t_{n+1} is smaller (in norm) than the solution at time t_n puts a limit on how large the macro-step size h, and consequently, how large the micro-step size h/m, can be: this is the condition of absolute stability. Specifically, absolute stability requires that the spectral radius of S_R is less than 1: $\rho(S_R) < 1$.

The stiffness of the system is quantified through the ratio M_{22}/M_{11} and the off-diagonal terms M_{12} and M_{21} represent the strength of coupling between the active

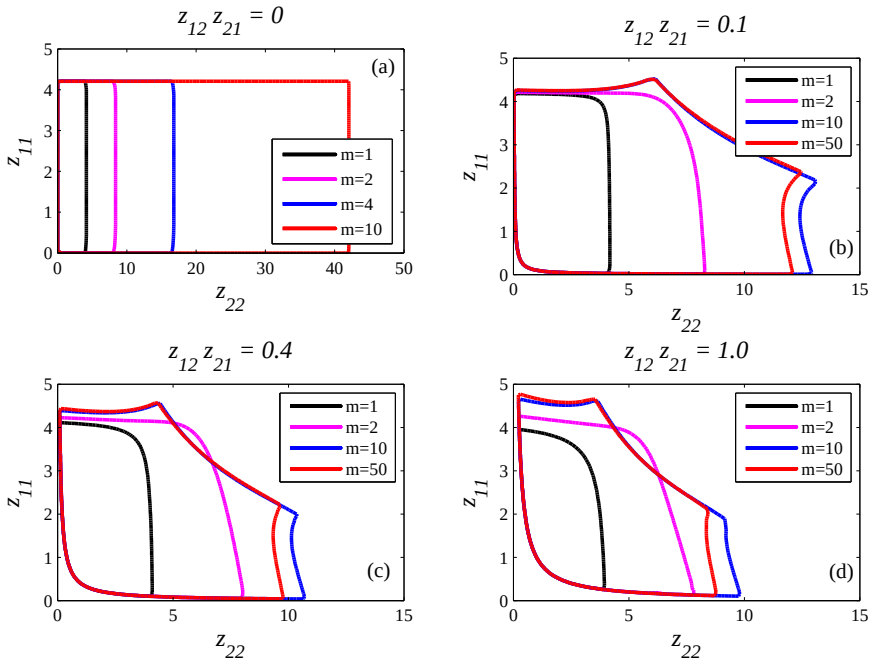

Fig. 3 Regions of absolute stability of the linear system (15) for different coupling strengths $z_{12}z_{21}$, where M takes the form (43). The $z_{ij} = hM_{ij}$ where h is the macro-step size.

and latent components. Defining $z_{ij} = hM_{ij}$, one can see from the definitions of ζ_1, ζ_2 in (30), (31) and L_i, U_{ij} in the appendix that

$$\begin{array}{lll} L_1 & = I_6 + z_{11}A - z_{12}z_{21}A(I_6 + z_{22}A)^{-1}A & = P_1(z_{11}, z_{22}, z_{12}z_{21}), \\ L_2 & = I_6 + z_{22}A - z_{12}z_{21}A(I_6 + z_{11}A)^{-1}A & = P_2(z_{11}, z_{22}, z_{12}z_{21}), \\ U_{11} & = -z_{11}\mathbf{1} + z_{12}z_{21}A(I_6 + z_{22}A)^{-1}\mathbf{1} & = P_3(z_{11}, z_{22}, z_{12}z_{21}), \\ U_{12} & = -z_{12}\mathbf{1} + z_{12}z_{22}A(I_6 + z_{22}A)^{-1}\mathbf{1} & = z_{12}P_4(z_{22}), \\ \zeta_1 & = -\frac{z_{21}}{m}\mathbf{b}^T\left[I_6 + \frac{z_{22}}{m}A\right]^{-1} & = \frac{z_{21}}{m}P_5\left(\frac{z_{22}}{m}\right), \\ \zeta_2 & = 1 - \frac{z_{22}}{m}\mathbf{b}^T\left[I_6 + \frac{z_{22}}{m}A\right]^{-1}\mathbf{1} & = P_6\left(\frac{z_{22}}{m}\right), \end{array} \tag{45}$$

for known functions P_j, $j = 1, \ldots, 6$. Using these forms in eqs. (34)-(37), we find that the elements of S_R are

$$S_{11} = 1 + \mathbf{b}^T P_1^{-1} P_3, \tag{46}$$

$$S_{12} = z_{12}\mathbf{b}^T P_1^{-1} P_4, \tag{47}$$

$$S_{21} = \frac{z_{21}}{m}\sum_{j=0}^{m-1} P_6^{m-j-1} P_5 \mathbf{1} + \frac{z_{21}}{m}\sum_{j=0}^{m-1} P_6^{m-j-1} P_5 \Omega(j) Q P_1^{-1} P_3, \tag{48}$$

$$S_{22} = P_6^m + \frac{z_{12}z_{21}}{m}\sum_{j=0}^{m-1} P_6^{m-j-1} P_5 \Omega(j) Q P_1^{-1} P_4, \tag{49}$$

where the arguments of each of the P_j have been suppressed, but are identical to the ones in (45).

The eigenvalues of S_R depend only on its trace and determinant. From (46)-(49) it is clear that both the trace $S_{11} + S_{22}$ and the determinant $S_{11}S_{22} - S_{12}S_{21}$ only depend on z_{11}, z_{22} and the product $z_{12}z_{21}$, which represents the strength of coupling between $x(t)$ and $y(t)$. Therefore the spectral radius of S_R depends only on z_{11}, z_{22} and the product $z_{12}z_{21}$. In Fig. 3, we plot the 1-level curves of the spectral radius to represent regions of absolute stability.

In Figure 3(a) when either of M_{12} or M_{21} are zero, we see that the regions of absolute stability are rectangular and the width of the region is proportional to m. When $M_{11} = O(1)$ but M_{22} is large (but fixed), the active component decays rapidly and the system is stiff: using a standard explicit solver would lead to instability. Figure 3(a) shows that the explicit method, in principle, can be stabilized by using small timesteps for the rapidly decaying component and choosing m sufficiently large. In Fig. 3(b)-(d), we see that for fixed coupling $z_{12}z_{21}$, increasing m increases the size of the region of absolute stability, but the regions do not grow indefinitely with m. Instead, as $m \to \infty$, there appears to be a limiting stability region. Interestingly, while increasing m mainly increases the width of the region of absolute stability, there is also a small increase in the height: the multirate extension can also have a stabilizing effect on the latent component. Our analysis also suggests that there is no significant gain in stability for $m \gtrsim 2$ when the coupling between active and latent components is strong. For fixed m, increasing the coupling strength $z_{12}z_{21}$ decreases the size of the absolute stability region. The effect is most significant when $z_{12}z_{21}$ increases from zero: for example compare Figs. 3(a) and (b). In this case, when $z_{12}z_{21}$ increases from 0 to 0.1 and $m = 10$, the diagonal entry M_{22} must reduce by about 1/3 in order to retain stability for fixed h.

4 Interpolation

The key to making a multirate method high order is in generating dense output from latent components with high accuracy. One way to do this is through interpolation.[1] For some integrators it is obvious how to derive an interpolant that is consistent with the underlying integrator. The Backward Differentiation Formulas (BDF), which use extrapolation from prior time steps to advance the solution in time, are an example of this. Interpolants for Runge-Kutta methods are discussed in [10].

We hypothesize that when the micro-integrator is explicit, interpolating the latent components using an $(n-1)$ order interpolant coupled to an integrator which is order n gives a multirate method that is order n overall. Certainly, we found that this is the case for the Cash-Karp method discussed in this paper ($n = 4$). One can also show that Heun's method (second order Runge Kutta) together with linear interpolation gives a second order multirate method.

In [12], the authors found that using linear interpolation reduced the order of a multirate (implicit) trapezoidal rule. However, the active components were fixed *a priori*, and so were the interpolated components. We think that the choice

[1] We use the word "interpolation" to describe a method to construct a smooth approximation to the numerical solution between two times t_n and t_{n+1}. However, the function that we derive does not pass through the numerical solution at t_{n+1}. Strictly speaking, it is not an interpolant. Nevertheless, we still refer to these approximating functions as "interpolants."

of which components to interpolate is important. We advocate that interpolation should be done for solution components that are "sufficiently slow," though it is not clear how to quantify this statement at present. What is clear is that the interpolant (14) (or any high order interpolant) could still give large interpolation errors that would contaminate the accuracy of the micro-integrator if $\mathbf{x}(t)$ had large derivatives.

Derivation of interpolants. We now illustrate how interpolants can be constructed during run-time for the Cash-Karp Runge-Kutta formula [16]. Our main result is the third order interpolant (14) which we believe is new. Suppose we have the simple scalar ODE $\dot{y} = f(t, y)$. Then to construct a third order interpolant, note that for $0 \leq \chi \leq 1$,

$$y(t_n + \chi h) = y_n + \chi h y'_n + \frac{(\chi h)^2}{2} y''_n + \frac{(\chi h)^3}{6} y'''_n + O(h^4). \tag{50}$$

By repeatedly differentiating the ODE $y'_n = f(t_n, y_n)$, we have $y'_n = G_0$, $y''_n = G_1$, $y'''_n = G_2 + G_3$, where $G_0 = f_n$, $G_1 = f_n f'_n + \dot{f}_n$, $G_2 = f_n^2 f''_n + 2 f_n \dot{f}'_n + \ddot{f}_n$ and $G_3 = f_n {f'_n}^2 + f'_n \dot{f}_n$. Our convention is to use a dash for a y-derivative and a dot for a t-derivative: $f'_n = \frac{\partial f(t_n, y_n)}{\partial y}$ and $\dot{f}_n = \frac{\partial f(t_n, y_n)}{\partial t}$. Therefore, (50) becomes

$$y(t_n + \chi h) = y_n + \chi h G_0 + \frac{(\chi h)^2}{2} G_1 + \frac{(\chi h)^3}{6} (G_2 + G_3) + O(h^4). \tag{51}$$

Using definitions $k_i = h f(t_n + c_i h, y_n + \sum_{j=1}^{i-1} a_{ij} k_j)$ and expanding in Taylor series, we have

$$\begin{aligned}
k_1 &= h G_0, \\
k_2 &= h G_0 + \frac{h^2}{5} G_1 + \frac{h^3}{50} G_2 + O(h^4), \\
k_3 &= h G_0 + \frac{3h^2}{10} G_1 + \frac{9h^3}{200} (G_2 + G_3) + O(h^4), \\
k_4 &= h G_0 + \frac{3h^2}{5} G_1 + \frac{9h^3}{50} (G_2 + G_3) + O(h^4), \\
k_5 &= h G_0 + h^2 G_1 + \frac{h^3}{2} (G_2 + G_3) + O(h^4), \\
k_6 &= h G_0 + \frac{7h^2}{8} G_1 + \frac{49h^3}{128} (G_2 + G_3) + O(h^4).
\end{aligned} \tag{52}$$

Taking only the equations for k_1, k_4 and k_5, we form a linear system in terms of G_0, G_1 and $G_2 + G_3$:

$$\begin{pmatrix} h & 0 & 0 \\ h & \frac{3h^2}{5} & \frac{9h^3}{50} \\ h & h^2 & \frac{h^3}{2} \end{pmatrix} \begin{pmatrix} G_0 \\ G_1 \\ G_2 + G_3 \end{pmatrix} = \begin{pmatrix} k_1 \\ k_4 \\ k_5 \end{pmatrix}. \tag{53}$$

Upon solving for G_0, G_1 and $G_2 + G_3$ and substituting into (51), we find

$$\begin{aligned}
y(t_n + \chi h) = y_n + \chi k_1 + \frac{\chi^2}{2} \left(-\frac{8}{3} k_1 + \frac{25}{6} k_4 - \frac{3}{2} k_5 \right) \\
+ \frac{\chi^3}{6} \left(\frac{10}{3} k_1 - \frac{25}{3} k_4 + 5 k_5 \right) + O(h^4).
\end{aligned} \tag{54}$$

A similar calculation can be performed for classical 4th order Runge-Kutta formulas and other embedded formulas (e.g. Dormand-Prince) to generate cubic interpolants. Unfortunately, this method fails when quartic (or higher order) interpolants are required. When we increase the accuracy of (51) to $O(h^5)$, the analogous matrix in eq. (53) is singular for all the RK formulas that we analyzed (Dormand-Prince, Cash-Karp, Fehlberg). However, quartic and higher order interpolants *can* be constructed if one includes more intermediate stage function evaluations in the Runge-Kutta formula [19].

5 Validation and Results

We now test our method on some large ODE systems that arise from a method-of-lines discretization of some common physical PDEs. Our examples also follow the sample problems that were explored in [12,18]. We use these simple test problems to illustrate proof of principle and demonstrate 4th order convergence of the method. Matlab codes used to generate the results in this section can be found on the author's homepage.

Example 1: Transport Equation. The problem considered is a method-of-lines discretization of $\frac{\partial y}{\partial t} + U\frac{\partial y}{\partial x} = 0$, $y(x,0) = \exp(-x^2)$, on the infinite domain $-\infty < x < \infty$. For $U > 0$ we use an upwind discretization

$$\dot{y}_1(t) = 0, \tag{55}$$

$$\dot{y}_i(t) = -\frac{U}{\Delta x}(y_i - y_{i-1}), \qquad i = 2, 3, \ldots, N, \tag{56}$$

with $x_i = -L + (i-1)\Delta x$, $\Delta x = 2L/(N-1)$ and $y_i(0) = \exp(-x_i^2)$, $i = 1, \ldots, N$ is the initial condition. The numerical domain for eqs. (55)-(56) is $[-L, L]$. Figure 4 shows the result of applying our multirate integration method to (55)-(56). The method flags components of the solution that move on a fast time scale, corresponding to regions in space which are perturbed by the travelling wave, and the flag "footprint" $[r, s]$ generally follows the traveling wave. The upwind discretization has a truncation error that corresponds to a diffusion term which is responsible for the spreading of the initial condition. This diffusion also ensures that the number of flagged components increases as time progresses.

Example 2: Reaction-Diffusion equation. We now consider a discretization of the reaction-diffusion system $\frac{\partial y}{\partial t} = \varepsilon\frac{\partial^2 y}{\partial x^2} + \gamma y^2(1-y)$, $y(x,0) = (1 + \exp[\lambda(x-1)])^{-1}$, with $\lambda = \frac{1}{2}\sqrt{2\gamma/\varepsilon}$, which admits a travelling wave solution. A centered-in-space discretization gives

$$\dot{y}_1(t) = 0, \tag{57}$$

$$\dot{y}_i(t) = \frac{\varepsilon}{\Delta x^2}(y_{i+1} - 2y_i + y_{i-1}) + \gamma y_i^2(1 - y_i), \qquad i = 2, 3, \ldots, N-1, \tag{58}$$

$$\dot{y}_N(t) = 0, \tag{59}$$

with $x_i = (i-1)\Delta x$, $\Delta x = L/(N-1)$ and $y_i(0) = (1 + \exp[\lambda(x_i - 1)])^{-1}$, $i = 1, \ldots, N$ is the initial condition. The numerical domain of solution is $[0, L]$. Figure 5 shows the results of applying our multirate integration method to the nonlinear

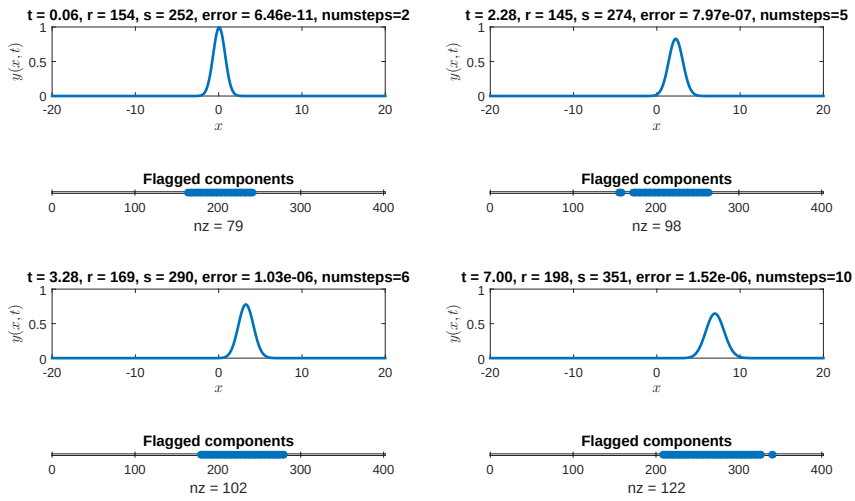

Fig. 4 Multirate integration of discretized transport equation (55)-(56). Indices of flagged components are indicated on a lattice below each profile. `numsteps` is the number of macrosteps taken; `nz` is the number of flagged components. Parameters were $L = 20$, $N = 401$, $\Delta x = 0.1$, $TOL = 10^{-6}$, $\delta = 10^{-4}$, $U = 1$.

ODE system (57)-(59). As with example 1, because the disturbance is localized in space, our method generally flags the components i that have large $\dot{y}_i$ while leaving the remaining components unflagged. The value of $\delta = 10^{-2}$ used here is large enough so that the number of flagged components is relatively small and flagged components correspond to x in the PDE where $\frac{\partial y}{\partial x}$ is large. For this example, we found that the flag footprint $[r, s]$ was very sensitive to δ: when δ was decreased, `nz` rapidly increased and the algorithm flagged many quiescent components where $y_i \approx 0$ or $y_i \approx 1$.

Example 3: Advection-Diffusion Equation. As a third example, we consider a discretization of a forced advection-diffusion equation $\frac{\partial y}{\partial t} + a\frac{\partial y}{\partial x} = d\frac{\partial^2 y}{\partial x^2} - cy + g(x,t)$, $g(x,t) = 1000\,[\cos(\pi x/2)]^{200}\sin(\pi t)$, $y(x,0) = 0$, on $-1 \le x \le 1$. We use a fourth order spatial discretization that results in a broader stencil and a pentadiagonal

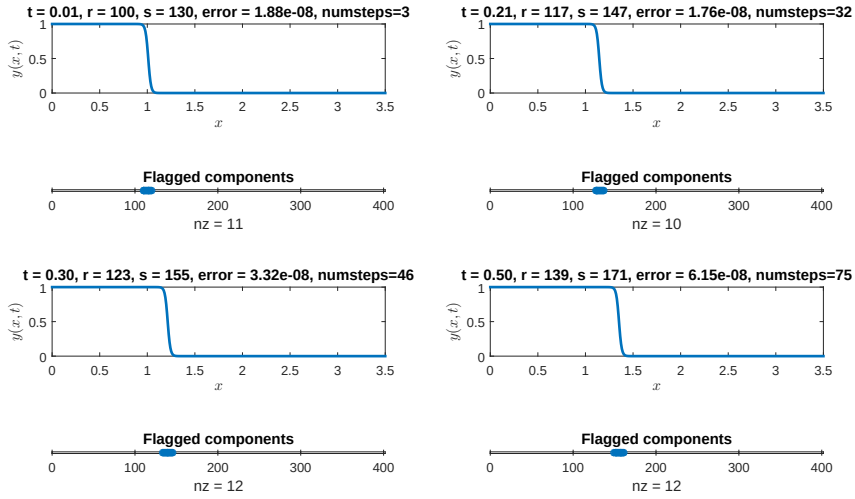

Fig. 5 Multirate integration of reaction-diffusion equation (57)-(59). See caption of Fig. 4 for explanation of `numsteps` and `nz`. Parameters were $L = 3.5$, $N = 401$, $\Delta x = 8.75 \times 10^{-3}$, $TOL = 10^{-8}$, $\delta = 10^{-2}$, $\varepsilon = 0.01$, $\gamma = 100$.

system of ODEs:

$$\dot{y}_1(t) = 0, \tag{60}$$

$$\dot{y}_2(t) = 0, \tag{61}$$

$$\dot{y}_i(t) = -\frac{a(-y_{i+2} + 8y_{i+1} - 8y_{i-1} + y_{i-2})}{12\Delta x} + \frac{d(-y_{i+2} + 16y_{i+1} - 30y_i + 16y_{i-1} - y_{i-2})}{12\Delta x^2} - cy_i + g(x_i, t), \quad i = 3, 4, \ldots, N-2, \tag{62}$$

$$\dot{y}_{N-1}(t) = 0, \tag{63}$$

$$\dot{y}_N(t) = 0, \tag{64}$$

with $x_i = -1 + (i-1)/\Delta x$ and $\Delta x = 2/(N-1)$ and $y_i(0) = 0, i = 1, 2, \ldots, N$ is the initial condition. Figure 6 shows the results of our method applied to the system (60)-(64), which are more mixed. On the one hand, the number of flagged components in this problem rapidly increases in time. At the end of the integration for $t = 0.80$, most of all $N = 401$ components are flagged even though the solution is localized in a small region of x. On the other hand, even though the system (60)-(64) is stiff, our method is able to complete the integration in 9 macro-steps even though it is based on explicit Runge-Kutta formulas. (In contrast, a single rate method takes many more macro-steps; see below.)

Next, we perform a convergence study to demonstrate that cubic interpolants coupling the macro and micro integrators together give a fourth order method.

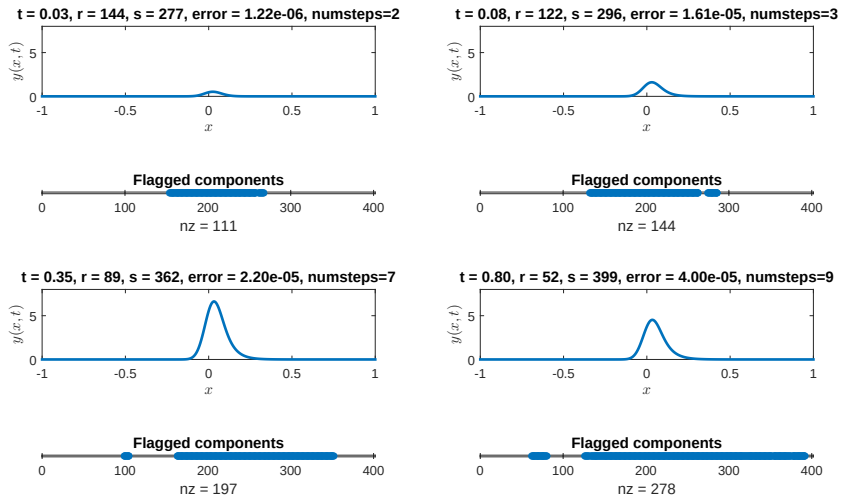

Fig. 6 Multirate integration of advection-diffusion equation (60)-(64). See caption of Fig. 4 for explanation of `numsteps` and `nz`. Parameters were $L = 1$, $N = 401$, $\Delta x = 5 \times 10^{-3}$, $TOL = 10^{-5}$, $\delta = 10^{-5}$, $a = 5$, $d = 0.01$, $c = 100$.

For this error analysis, we implement our method in fixed timestep mode: h is constant throughout the integration and we take $m = 10$. We also fix the values of r and s so that the same components are treated as active throughout the whole course of the integration and the error is only computed for components r to s. The "exact" solution is found as a superposition of normal modes (with eigenvectors and eigenvalues determined numerically) in the transport equation and is computed using MATLAB's ODE solvers, set to a suitably stringent tolerance, for the reaction-diffusion and advection-diffusion equations. Figure 7 confirms fourth order convergence in terms of the infinity norm of the error.

Finally, it is instructive to compare single rate and multirate methods. Ideally, for a given macro-step tolerance, we would like our multirate method to take fewer macro-steps than a single rate method, but incur roughly the same error at the end of the integration (in a single rate method, the normal time step is considered the macro-step and there are no micro-steps). We compare the number of steps taken and the errors incurred for all three test problems in Table 2.

Our multirate method always takes fewer macro-steps than the single rate method. For all three systems of equations, the global error incurred is roughly the same whether we use a single rate or multirate method. These results are largely independent of TOL and δ. For much smaller TOL we did find that the multirate error could be a few orders of magnitude smaller than the single rate error. The implication is that the step size used by the micro-integrator is too small. In this case, it is desirable to find a way to increase the size of the micro-steps to give a more efficient multirate method.

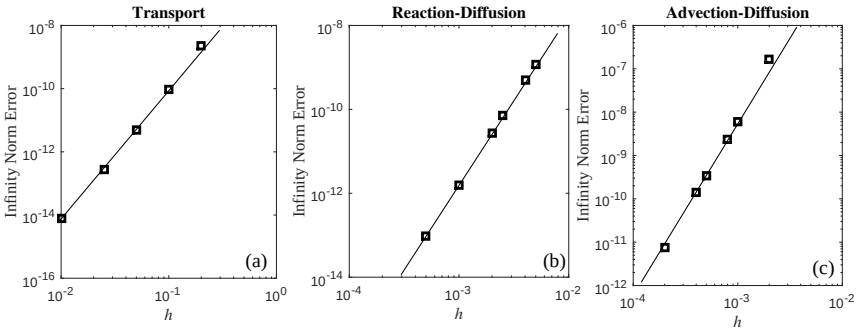

Fig. 7 Convergence of Cash-Karp multirate method in fixed timestep mode, applied to each of the three examples above with $TOL = 10^{-6}$ and $\delta = 10^{-12}$. Here h is the size of the macro-step and there are $m = 10$ micro-steps per macro-step. Solid lines have slope 4. The active components are $[r, s] = [186, 216], [93, 139], [201, 241]$ for (a), (b) and (c) respectively with $N = 401$ equations in each case, corresponding to the active components lying in $-1.5 \leq x \leq 1.5$, $0.805 \leq x \leq 1.2075$ and $0 \leq x \leq 0.2$. The final integration times were $T = 7$ for (a), $T = 0.5$ for (b) and $T = 0.8$ for (c).

Equation	# macro-steps (error)	# macro-steps/# micro-steps (error)
Transport	30 (9.00×10^{-5})	10/40 (6.03×10^{-5})
Reaction-Diffusion	74 (2.14×10^{-5})	21/144 (4.50×10^{-7})
Advection-Diffusion	409 (1.11×10^{-5})	5/420 (7.95×10^{-6})

Table 2 Comparison of single rate (second column) and multirate (third column) integration methods in terms of number of steps taken and error incurred. Error was measured using the infinity norm and the code parameters were $TOL = 10^{-4}$ and $\delta = 10^{-12}$. The initial step sizes for the transport, reaction-diffusion and advection-diffusion equations were $\Delta t = 10^{-2}, 5\times10^{-4}$ and 5×10^{-3} for both single and multirate methods.

6 Conclusions

In this paper, we presented a fourth order linearly accurate multirate method. The method is based on embedded Runge-Kutta (RK) formulas and is explicit in time. The main strength of the method is its high order; we showed in this paper that only third order interpolants are needed for a fourth order solver. Our algorithm generates these interpolants by using the intermediate stage function evaluations of an embedded RK formula. We tested our method on three simple problems and performed a convergence study to confirm the method order.

We see three main extensions to this work. The first is to understand mathematically the spread of the active components in time. Generally, we found that our method is effective when the solutions are wave-like and disturbances are localized in index-space, but not so competitive when they are diffusive (compare Figures 4 and 5 with Figure 6). We would like the number of flagged components to remain small relative to the total number of components, throughout the course of the integration (in the extreme case when $r = 1$ and $s = N$, all components are being integrated twice and one should revert to a single rate method). However

in practice this can be difficult to achieve and the fraction of flagged components seems to be sensitive to δ and the governing system of equations.

The second is to extend our method to other RK formulas. The interpolant used in our method is derived from a Taylor series at t_n and its order stems from derivative information at t_n which in turn comes from intermediate stage function evaluations. However, there are many other methods available to build interpolants and this is mature area of research: for example, see the techniques discussed in [10]. The control of interpolation errors and how to to choose the interpolating components should remain active areas of study within the field of multirate integration.

Finally, there is the issue of how to choose the parameter δ. In our method, a smaller δ results in more components being flagged as active. Although this may seem wasteful, latent components are able to advance with a larger timestep. Conversely, a larger δ results in fewer components being flagged as active, but latent components have to advance with a smaller timestep. It is not clear at present which value of δ results in an efficient multirate method, or how to determine good macro-step and micro-step sizes from the governing system of equations.

In summary, this work contributes to the currently growing body of research in multirate methods. We hope that the strategies presented in this paper can be extended to the numerical solution of other physical problems and be used to improve the efficiency and accuracy of other multirate algorithms.

Acknowledgements: The author thanks R. R. Rosales who stimulated the author's interest in multirate methods and proposed the derivation of cubic interpolants.

A Matrix Definitions

Here we given the definitions of the matrices in eqs. (17)-(19).

$$\begin{aligned}
L_0 &= I_{6N_0} + h(A \otimes M_{00}) - h^2(A \otimes M_{01})\,[\Sigma_2(M, hA)]^{-1}\,(A \otimes M_{10}),\\
L_1 &= -I_{6N_1} - h(A \otimes M_{11}) + \Sigma_0(M, hA) + \Sigma_2(M, hA),\\
L_2 &= I_{6N_2} + h(A \otimes M_{22}) - h^2(A \otimes M_{21})\,[\Sigma_0(M, hA)]^{-1}\,(A \otimes M_{12}),
\end{aligned}$$

$$\begin{aligned}
U_{00} &= -h(\mathbf{1} \otimes M_{00}) + h^2(A \otimes M_{01})\,[\Sigma_2(M, hA)]^{-1}\,(\mathbf{1} \otimes M_{10}),\\
U_{01} &= -h(\mathbf{1} \otimes M_{01}) + h^2(A \otimes M_{01})\,[\Sigma_2(M, hA)]^{-1}\,(\mathbf{1} \otimes M_{11})\\
&\quad -h^3(A \otimes M_{01})\,[\Sigma_2(M, hA)]^{-1}\,(A \otimes M_{12})(I_{6N_2} + hA \otimes M_{22})^{-1}(\mathbf{1} \otimes M_{21}),\\
U_{02} &= h^2(A \otimes M_{01})\,[\Sigma_2(M, hA)]^{-1}\,(\mathbf{1} \otimes M_{12})\\
&\quad -h^3(A \otimes M_{01})\,[\Sigma_2(M, hA)]^{-1}\,(A \otimes M_{12})(I_{6N_2} + hA \otimes M_{22})^{-1}(\mathbf{1} \otimes M_{22}),
\end{aligned}$$

$$\begin{aligned}
U_{10} &= -h(\mathbf{1} \otimes M_{10}) + h^2(A \otimes M_{10})(I_{6N_0} + hA \otimes M_{00})^{-1}(\mathbf{1} \otimes M_{00}),\\
U_{11} &= -h(\mathbf{1} \otimes M_{11}) + h^2(A \otimes M_{10})(I_{6N_0} + hA \otimes M_{00})^{-1}(\mathbf{1} \otimes M_{01})\\
&\quad +h^2(A \otimes M_{12})(I_{6N_2} + hA \otimes M_{22})^{-1}(\mathbf{1} \otimes M_{21}),\\
U_{12} &= -h(\mathbf{1} \otimes M_{12}) + h^2(A \otimes M_{12})(I_{6N_2} + hA \otimes M_{22})^{-1}(\mathbf{1} \otimes M_{22}).
\end{aligned}$$

$$
\begin{aligned}
U_{20} &= h^2(A\otimes M_{21})\left[\Sigma_0(M,hA)\right]^{-1}(\mathbf{1}\otimes M_{10})\\
&\quad -h^3(A\otimes M_{21})\left[\Sigma_0(M,hA)\right]^{-1}(A\otimes M_{10})(I_{6N_0}+hA\otimes M_{00})^{-1}(\mathbf{1}\otimes M_{00}),\\
U_{21} &= -h(\mathbf{1}\otimes M_{21})+h^2(A\otimes M_{21})\left[\Sigma_0(M,hA)\right]^{-1}(\mathbf{1}\otimes M_{11})\\
&\quad -h^3(A\otimes M_{21})\left[\Sigma_0(M,hA)\right]^{-1}(A\otimes M_{10})(I_{6N_0}+hA\otimes M_{00})^{-1}(\mathbf{1}\otimes M_{01}),\\
U_{22} &= -h(\mathbf{1}\otimes M_{22})+h^2(A\otimes M_{21})\left[\Sigma_0(M,hA)\right]^{-1}(\mathbf{1}\otimes M_{12})
\end{aligned}
$$

where for matrices Z and $M=\begin{bmatrix} M_{00} & M_{01} & M_{02}\\ M_{10} & M_{11} & M_{12}\\ M_{20} & M_{21} & M_{22}\end{bmatrix}$, we define

$$
\begin{aligned}
\Sigma_0(M,Z) &= I_{6N_1}+(Z\otimes M_{11})-(Z\otimes M_{10})(I_{6N_0}+Z\otimes M_{00})^{-1}(Z\otimes M_{01}),\\
\Sigma_2(M,Z) &= I_{6N_1}+(Z\otimes M_{11})-(Z\otimes M_{12})(I_{6N_2}+Z\otimes M_{22})^{-1}(Z\otimes M_{21}).
\end{aligned}
$$

References

1. J. F. Andrus. Numerical solution of systems of ordinary differential equations separated into subsystems. *SIAM J. Numer. Anal.*, 16: 605-611, 1979.
2. A. Bartel and M. Günther. A multirate W-method for electrical networks in state-space formulation. *J. Comp. Appl. Math.*, 147(2):411–425, 2002.
3. A. Bartel, M. Günther and A. Kvaerno. Multirate methods in electrical circuit simulation. *Progress in Industrial Mathematics at ECMI 2000*, 258-265, 2000.
4. E. M. Constantinescu and Adrian Sandu. Extrapolated multirate methods for differential equations with multiple time scales, *Journal of Scientific Computing*, 2012.
5. E. M. Constantinescu and Adrian Sandu. Multirate Timestepping Methods for Hyperbolic Conservation Laws *Journal of Scientific Computing*, 33: 239–278, 2007.
6. E. M. Constantinescu and Adrian Sandu. Multirate Explicit Adams Methods for Time Integration *Journal of Scientific Computing*, 38: 229–249, 2009.
7. W. H. Enright, K. R. Jackson, S. P. Nørsett, and P. G. Thomsen. Interpolants for Runge-Kutta formulas. *ACM Trans. Math. Softw.*, 12(3):193–218, 1986.
8. C. W. Gear and D. R. Wells. Multirate linear multistep methods. *BIT*, 24:484–502, 1984.
9. M. Günther, A. Kværnø, and P. Rentrop. Multirate partitioned Runge-Kutta methods. *BIT*, 41(3):504–514, 2001.
10. E. Hairer and S.P. Norsett and G. Wanner. Solving Ordinary Differential Equations I: Nonstiff Problems. *Springer Series in Computational Mathematics*, 1987
11. M. K. Horn. Fourth and fifth order, scaled Runge-Kutta algorithms for treating dense output. *SIAM J. Numer. Anal.*, 20(3):558–568, 1983.
12. W. Hundsdorfer and V. Savcenco. Analysis of a multirate theta-method for stiff ODEs. *Applied Numerical Mathematics*, 59:693–706, 2009.
13. T. Kato and T. Kataoka. Circuit analysis by a new multirate method. *Electr. Eng. Jpn.*, 126(4):1623–1628, 1999.
14. A. Logg. Multi-adaptive time integration. *Appl. Numer. Math.*, 48:339–354, 2004.
15. J. Makino and S. Aarseth. On a Hermite integrator with Ahmad-Cohen scheme for gravitational many-body problems. *Publ. Astron. Soc. Japan*, 44:141–151, 1992.
16. W. H. Press, S. A. Teukolsky, W. V. Vetterling, and B. P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, second edition, 1992.
17. V. Savcenco. Construction of a multirate RODAS method for stiff ODEs *Journal Comp. Appl. Math.*, 225: 323-337, 2009.
18. V. Savcenco, W. Hundsdorfer, and J. G. Verwer. A multirate time stepping strategy for stiff ordinary differential equations. *BIT*, 47:137–155, 2007.
19. L. F. Shampine. Some Practical Runge-Kutta Formulas *Mathematics of Computation*, 173: 135-150, 1986.
20. S. Skelboe. Stability properties of backward differentiation multirate formulas. *Appl. Numer. Math.*, 5: 151-160, 1989.
21. J. Waltz, G. L. Page, S. D. Milder, J. Wallin, and A. Antunes. A performance comparison of tree data structures for N-body simulation. *J. Comp. Phys.*, 178:1–14, 2002.