Numerical Results: Multiple time stepping algorithms for explicit one-step exponential integrators

May 13, 2018

1 Problem definition

Goal of paper:
Construct efficient (MTS) algorithms based on classes of explicit one-step exponential integrators.

What's desirable about these methods?

- Application to exponential Rosenbrock methods
- Do not require computation of starting values
- Easy implementation of adaptive time step control
- Avoid matrix functions
- Order of accuracy up to four to be demonstrated

Problem we're solving:

$$u'(t) = F(t, u(t)) = Au(t) + g(t, u(t)), u(t_0) = u_0 \tag{1.1}$$

on the interval $t_0 \leq t \leq T$, where the vector field $F(t, u(t))$ can be decomposed into a linear stiff part $Au(t)$ and nonlinear nonstiff part $g(t, u(t))$. We consider the case where the stiff part is cheap to compute and the nonstiff part is expensive.

1.1 MTS Algorithm for ETD Methods

$$\hat{p}_{n,i}(\tau) = \sum_{j=1}^{i-1}\left(\sum_{k=1}^{l_{ij}} \frac{\alpha_{ij}^{(k)}}{c_i^k h^{k-1}(k-1)!}\tau^{k-1}\right) g(t_n + c_j h, \hat{U}_{n,j}), \tag{3.19a}$$

$$\hat{q}_{n,s}(\tau) = \sum_{i=1}^{s}\left(\sum_{k=1}^{m_i} \frac{\beta_i^{(k)}}{h^{k-1}(k-1)!}\tau^{k-1}\right) g(t_n + c_i h, \hat{U}_{n,i}), \tag{3.19b}$$

To get $\hat{u}_{n+1}$ solve:

$$y'_{n,i}(\tau) = A y_{n,i}(\tau) + \hat{p}_{n,i}(\tau), \quad y_{n,i}(0) = \hat{u}_n \quad (2 \le i \le s) \tag{3.20}$$

1.1.1 Strategy for computing $\hat{U}_{n,i}$

1. Set $\hat{U}_{n,1} = \hat{u}_n$.
2. $\hat{U}_{n,1}$ is now known. Evaluate $\hat{p}_{n,2}(\tau)$ from (3.19a) and solve (3.20) with $i = 2$ to obtain $\hat{U}_{n,2} \approx \hat{y}_{n,2}(c_2 h)$.
3. $\hat{U}_{n,1}, \hat{U}_{n,2}$ is now known. Evaluate $\hat{p}_{n,3}(\tau)$ from (3.19a) and solve (3.20) with $i = 3$ to obtain $\hat{U}_{n,3} \approx \hat{y}_{n,3}(c_3 h)$.
   $\vdots$
4. $\hat{U}_{n,1}, \cdots \hat{U}_{n,s-1}$ is now known. Evaluate $\hat{p}_{n,s}(\tau)$ from (3.19a) and solve (3.20) with $i = s$ to obtain $\hat{U}_{n,s} \approx \hat{y}_{n,s}(c_s h)$.

$$y'_n(\tau) = A y_n(\tau) + \hat{q}_n(\tau), \quad y_n(0) = \hat{u}_n \tag{3.21}$$

2 Numerical Algorithms

2.1 A third order method `expRK32`

Consider the three stage ETD method of stiff order three with the following Butcher tableau.

$$\begin{array}{c|ccc}
0 & & & \\
c_2 & c_2\varphi_{1,2} & & \\
\frac{2}{3} & \frac{2}{3}\varphi_{1,3} - \frac{4}{9c_2}\varphi_{2,3} & \frac{4}{9c_2}\varphi_{2,3} & \\
\hline
 & \varphi_1 - \frac{3}{2}\varphi_2 & 0 & \frac{3}{2}\varphi_2 \\
 & \varphi_1 & \frac{3}{3c_2-2}\varphi_2 & \frac{3}{2-3c_2}\varphi_2
\end{array}$$

From the Butcher tableau,
$l_{21} = q_1 = 1$,
$l_{31} = l_{32} = m_1 = m_3 = q_2 = q_3 = 2$,
$\alpha_{21}^{(1)} = c_2, \alpha_{31}^{(1)} = \frac{2}{3}, \alpha_{32}^{(1)} = 0$,
$\beta_1^{(1)} = 1, \beta_1^{(2)} = -\frac{3}{2}, \beta_2^{(k)} = 0 \;\; \forall k = 1, ..., m_2$,
$\beta_3^{(1)} = 0, \beta_3^{(2)} = \frac{3}{2}$,
$\bar{\beta}_2^{(1)} = \bar{\beta}_3^{(1)} = 0$,
$\bar{\beta}_2^{(2)} = -\bar{\beta}_3^{(2)} = \frac{3}{3c_2-2}(c_2 \neq \frac{2}{3})$.

We then have:

$$\hat{p}_{n,2}(\tau) = g(t_n, \hat{u}_n), \tag{3.30a}$$

$$\hat{p}_{n,3}(\tau) = \frac{1}{c_2 h} g(t_n + c_2 h, \hat{U}_{n,2})\tau + (1 - \frac{\tau}{c_2 h})\hat{p}_{n,2}(\tau), \tag{3.30b}$$

$$\hat{q}_{n,3}(\tau) = \frac{3}{2h} g(t_n + \frac{2}{3}h, \hat{U}_{n,3})\tau + (1 - \frac{3\tau}{2h})\hat{p}_{n,2}(\tau), \tag{3.30c}$$

$$\hat{\hat{q}}_{n,3}(\tau) = \frac{3}{(3c_2 - 2)h}(g(t_n + c_2 h, \hat{U}_{n,2}) - g(t_n + \frac{2}{3}h, \hat{U}_{n,3}))\tau + \hat{p}_{n,2}(\tau). \tag{3.30c}$$

2.2 Third order method `expRK32s3`

$$\begin{array}{c|ccc}
c_1 = 0 & & & \\
c_2 \neq \frac{2}{3} & c_2\varphi_{1,2} & & \\
c_3 = \frac{2}{3} & \frac{2}{3}\varphi_{1,3} - \frac{4}{9c_2}\varphi_{2,3} & \frac{4}{9c_2}\varphi_{2,3} & \\
\hline
 & b_1 = \varphi_1 - b_2 - b_3 & b_2 = \frac{-2}{3c_2(c_2-\frac{2}{3})}\varphi_2 + \frac{2}{c_2(c_2-\frac{2}{3})}\varphi_3 & b_3 = \frac{c_2}{\frac{2}{3}(c_2-\frac{2}{3})}\varphi_2 - \frac{2}{\frac{2}{3}(c_2-\frac{2}{3})}\varphi_3 \\
 & \bar{b}_1 = \varphi_1 - \bar{b}_2 - \bar{b}_3 & \bar{b}_2 = \frac{-2}{3c_2(c_2-\frac{2}{3})}\varphi_2 & \bar{b}_3 = \frac{c_2}{\frac{2}{3}(c_2-\frac{2}{3})}\varphi_2
\end{array}$$

Note : $\varphi_{i,j} = \varphi_i(c_j hA)$, $b_i = b_i(hA)$.

<u>**Polynomials**</u>

Let

$$D_{n,i} = g(t_n + c_i h, \hat{U}_{n,i}) - g(t_n, \hat{u}_n). \tag{1}$$

$$\hat{p}_{n,2}(\tau) = g(t_n, \hat{u}_n) \tag{2}$$

$$\hat{p}_{n,3}(\tau) = g(t_n, \hat{u}_n) + \frac{4}{9c_2}\frac{\tau}{c_3^2 h} D_{n,2} \tag{3}$$

$$\hat{q}_{n,3}(\tau) = g(t_n, \hat{u}_n) + \frac{\tau}{h}\left(\frac{-2}{3c_2(c_2 - \frac{2}{3})}D_{n,2} + \frac{c_2}{\frac{2}{3}(c_2 - \frac{2}{3})}D_{n,3}\right) + \frac{\tau^2}{2h^2}\left(\frac{2}{c_2(c_2 - \frac{2}{3})}D_{n,2} - \frac{2}{\frac{2}{3}(c_2 - \frac{2}{3})}D_{n,3}\right) \tag{4}$$

$$\hat{\hat{q}}_{n,3}(\tau) = g(t_n, \hat{u}_n) + \frac{\tau}{h}\left(\frac{-2}{3c_2(c_2-\frac{2}{3})}D_{n,2} + \frac{c_2}{\frac{2}{3}(c_2-\frac{2}{3})}D_{n,3}\right) \tag{5}$$

2.3 Fourth order method `expRK43s6`

0					
c_2					
c_3	$a_{32} = \frac{c_3^2}{c_2}\varphi_{2,3}$				
c_4	$a_{42} = \frac{c_4^2}{c_2}\varphi_{2,4}$	$a_{4,3} = 0$			
c_5	$a_{52} = 0$	$a_{53} = \frac{-c_4 c_5^2}{c_3(c_3-c_4)}\varphi_{2,5} + \frac{2c_5^3}{c_3(c_3-c_4)}\varphi_{3,5}$	$a_{54} = \frac{c_3 c_5^2}{c_4(c_3-c_4)}\varphi_{2,5} - \frac{2c_5^3}{c_4(c_3-c_4)}\varphi_{3,5}$		
c_6	$a_{62} = 0$	$a_{63} = \frac{-c_4 c_6^2}{c_3(c_3-c_4)}\varphi_{2,6} + \frac{2c_6^3}{c_3(c_3-c_4)}\varphi_{3,6}$	$a_{64} = \frac{c_3 c_6^2}{c_4(c_3-c_4)}\varphi_{2,6} + \frac{2c_6^3}{c_4(c_3-c_4)}\varphi_{3,6}$	$a_{65} = 0$	
	$b_2 = 0$	$b_3 = 0$	$b_4 = 0$	$b_5 = \frac{-c_6}{c_5(c_5-c_6)}\varphi_2 + \frac{2}{c_5(c_5-c_6)}\varphi_3$	$b_6 = \frac{c_5}{c_6(c_5-c_6)}\varphi_2 - \frac{2}{c_6(c_5-c_6)}\varphi_3$
	$\bar{b}_2 = 0$	$\bar{b}_3 = \frac{-c_4}{c_3(c_3-c_4)}\varphi_2 + \frac{2}{c_3(c_3-c_4)}\varphi_3$	$\bar{b}_4 = \frac{c_3}{c_4(c_3-c_4)}\varphi_2 - \frac{2}{c_4(c_3-c_4)}\varphi_3$	$\bar{b}_5 = 0$	$\bar{b}_6 = 0$

Conditions to be satisfied: $c_3 \neq c_4, c_5 \neq c_6, c_6 \neq \frac{2}{3}, c_5 = \frac{4c_6-3}{6c_6-4}$.
Two sets of c values:

- $c_2 = \frac{1}{2} = c_3 = c_5; c_4 = \frac{1}{3}; c_6 = 1.$
- $c_2 = c_3 = \frac{1}{2}; c_4 = c_6 = \frac{1}{3}; c_5 = \frac{5}{6}.$

Polynomials

$$\hat{p}_{n,2}(\tau) = g(t_n, \hat{u}_n) \tag{6}$$

$$\hat{p}_{n,3}(\tau) = g(t_n, \hat{u}_n) + \frac{\tau}{c_2 h}D_{n,2} \tag{7}$$

$$\hat{p}_{n,4}(\tau) = g(t_n, \hat{u}_n) + \frac{\tau}{c_2 h}D_{n,2} \tag{8}$$

$$\hat{p}_{n,5}(\tau) = g(t_n, \hat{u}_n) + \frac{\tau}{h}\left(\frac{-c_4}{c_3(c_3-c_4)}D_{n,3} + \frac{c_3}{c_4(c_3-c_4)}D_{n,4}\right) + \frac{\tau^2}{2h^2}\left(\frac{2}{c_3(c_3-c_4)}D_{n,3} - \frac{2}{c_4(c_3-c_4)}D_{n,4}\right) \tag{9}$$

$$\hat{p}_{n,6}(\tau) = g(t_n, \hat{u}_n) + \frac{\tau}{h}\left(\frac{-c_4}{c_3(c_3-c_4)}D_{n,3} + \frac{c_3}{c_4(c_3-c_4)}D_{n,4}\right) + \frac{\tau^2}{2h^2}\left(\frac{2}{c_3(c_3-c_4)}D_{n,3} - \frac{2}{c_4(c_3-c_4)}D_{n,4}\right) \tag{10}$$

$$\hat{q}_{n,6}(\tau) = g(t_n, \hat{u}_n) + \frac{\tau}{h}\left(\frac{-c_6}{c_5(c_5-c_6)}D_{n,5} + \frac{c_5}{c_6(c_5-c_6)}D_{n,6}\right) + \frac{\tau^2}{2h^2}\left(\frac{2}{c_5(c_5-c_6)}D_{n,5} - \frac{2}{c_6(c_5-c_6)}D_{n,6}\right) \tag{11}$$

$$\hat{\hat{q}}_{n,6}(\tau) = g(t_n, \hat{u}_n) + \frac{\tau}{h}\left(\frac{-c_4}{c_3(c_3-c_4)}D_{n,3} + \frac{c_3}{c_4(c_3-c_4)}D_{n,4}\right) + \frac{\tau^2}{2h^2}\left(\frac{2}{c_3(c_3-c_4)}D_{n,3} - \frac{2}{c_4(c_3-c_4)}D_{n,4}\right) \tag{12}$$

3 Test problems

3.1 Brusselator Problem

We consider the Brusselator problem represented by *(Why?)*

$$\frac{\mathbf{dy}}{\mathbf{dt}} = \begin{bmatrix} a-(y_3+1)y_1+y_1^2y_2 \\ y_3y_1-y_1^2y_2 \\ \frac{b-y_3}{\epsilon}-y_1y_3 \end{bmatrix} \tag{13}$$

This equation in the form of (1.1) becomes

$$\frac{\mathbf{dy}}{\mathbf{dt}} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ \frac{-1}{\epsilon} & 0 & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} + \begin{bmatrix} a-(y_3+1)y_1+y_1^2y_2 \\ y_3y_1-y_1^2y_2 \\ \frac{b}{\epsilon}-y_1y_3 \end{bmatrix} \tag{14}$$

where $y_1(0) = 1.2, y_2(0) = 3.1$, and $y_3(0) = 3$, with parameters $a = 1, b = 3.5$, and $\epsilon = 0.01$. We evaluate over the time interval $[0, 10]$. We investigate the performance of the algorithm when ODE solvers of different orders are used for the inner steps. For each ODE solver, the macro time steps are given by the $h-$values and h/m determines the micro time steps.

3.2 Modified Prothero-Robinson Problem

We consider the Modified Prothero-Robinson Problem

$$\frac{\mathbf{dy}}{\mathbf{dt}} = \begin{bmatrix} \Gamma & \epsilon \\ \epsilon & 1 \end{bmatrix} \begin{bmatrix} y_1-g(t) \\ y_2-g(\omega t) \end{bmatrix} - \begin{bmatrix} g(t) \\ g(\omega t) \end{bmatrix}' \tag{15}$$

In the form of (1.1) we have

$$\frac{\mathbf{dy}}{\mathbf{dt}} = \begin{bmatrix} \Gamma & \epsilon \\ \epsilon & 1 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} - \begin{bmatrix} \Gamma & \epsilon \\ \epsilon & 1 \end{bmatrix} \begin{bmatrix} g(t) + g'(t) \\ g(\omega t) + g'(\omega t) \end{bmatrix}. \tag{16}$$

we set $g(t) = \cos(t)$ with $\omega = 100, \Gamma = -1/\omega = -0.01, \epsilon = 1/\omega = 0.01$.

3.3 Reaction-Diffusion problem

We consider a reaction-diffusion problem with a traveling wave solution.

$$u_t = \epsilon u_{xx} + \gamma u^2(1-u), \tag{17}$$

for $0 < x < L, 0 < t \leq T$. Initial and boundary conditions are given by

$$u_x(0,t) = u_x(L,t) = 0, \qquad u(x,0) = (1 + e^{\lambda(x-1)})^{-1}, \tag{18}$$

where $\lambda = \frac{1}{2}\sqrt{2\gamma/\epsilon}$.

3.4 Set up of test problems and results

Things we are keeping track of:

- Max error
- Time for evaluation
- Number of function calls

Parameters we can change

- h values (make them smaller and smaller) (Which ones?)
- m values (increase value), this translates to a change in h_{fast} (decrease value). (Which ones?)
- Inner ERK methods being used (Heun, ERK-3-3, ERK-4-4)

Questions:

- What's the strategy for setting up these tests? Need to know what drivers to create and what information to set up to be stored in each case.
- Nail down time intervals to solve problems on