Implicit Two-Derivative Runge-Kutta Methods

Angela Tsai
(joint work with Shixiao Wang and Robert Chan)

Department of Mathematics
The University of Auckland

SciCADE 2011, Toronto, Canada
11-15 July

Outline of Topics

Basic Background

- Two-derivative Runge-Kutta (TDRK) methods belong to the family of multi-derivative Runge-Kutta methods – they are one-step multi-stage methods.
- We consider an autonomous ODE system $y'(t) = f(y)$ with initial condition $y_0 = y(t_0)$ and known second derivative $y''(t) = f'(y)f(y) =: g(y)$.
- Numerical Scheme:

$$Y_i = y_n + h\sum_{j=1}^{s} a_{ij} f(Y_j) + h^2 \sum_{j=1}^{s} \widehat{a}_{ij} g(Y_j), \quad i = 1, \ldots, s,$$

$$y_{n+1} = y_n + h\sum_{i=1}^{s} b_i f(Y_i) + h^2 \sum_{i=1}^{s} \widehat{b}_i g(Y_i).$$

Basic Background

- Two-derivative Runge-Kutta (TDRK) methods belong to the family of multi-derivative Runge-Kutta methods – they are one-step multi-stage methods.
- We consider an autonomous ODE system $y'(t) = f(y)$ with initial condition $y_0 = y(t_0)$ and known second derivative $y''(t) = f'(y)f(y) =: g(y)$.
- Numerical Scheme:

$$Y_i = y_n + h\sum_{j=1}^{s} a_{ij} f(Y_j) + h^2 \sum_{j=1}^{s} \widehat{a}_{ij} g(Y_j), \quad i = 1, \ldots, s,$$

$$y_{n+1} = y_n + h\sum_{i=1}^{s} b_i f(Y_i) + h^2 \sum_{i=1}^{s} \widehat{b}_i g(Y_i).$$

Basic Background

- Two-derivative Runge-Kutta (TDRK) methods belong to the family of multi-derivative Runge-Kutta methods – they are one-step multi-stage methods.
- We consider an autonomous ODE system $y'(t) = f(y)$ with initial condition $y_0 = y(t_0)$ and known second derivative $y''(t) = f'(y)f(y) =: g(y)$.
- Numerical Scheme:

$$Y_i = y_n + h\sum_{j=1}^{s} a_{ij} f(Y_j) + h^2 \sum_{j=1}^{s} \widehat{a}_{ij} g(Y_j), \quad i = 1, \dots, s,$$

$$y_{n+1} = y_n + h\sum_{i=1}^{s} b_i f(Y_i) + h^2 \sum_{i=1}^{s} \widehat{b}_i g(Y_i).$$

Basic Background

- In a non-autonomous system, the variable t can be treated as a component of the y vector.
- Block Matrix Form:

$$Y = e \otimes y_n + h(A \otimes I_N)F(Y) + h^2(\widehat{A} \otimes I_N)G(Y),$$
$$y_{n+1} = y_n + h(b^T \otimes I_N)F(Y) + h^2(\widehat{b}^T \otimes I_N)G(Y),$$

where $e = [1]_{s\times 1}$, $A = [a_{ij}]_{s\times s}$, $\widehat{A} = [\widehat{a}_{ij}]_{s\times s}$, $b = [b_i]_{s\times 1}$, $\widehat{b} = [\widehat{b}_i]_{s\times 1}$, and

$$Y = \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_s \end{bmatrix}, \quad F(Y) = \begin{bmatrix} f(Y_1) \\ f(Y_2) \\ \vdots \\ f(Y_s) \end{bmatrix}, \quad G(Y) = \begin{bmatrix} g(Y_1) \\ g(Y_2) \\ \vdots \\ g(Y_s) \end{bmatrix}.$$

Basic Background

- In a non-autonomous system, the variable t can be treated as a component of the y vector.
- Block Matrix Form:

$$Y = e \otimes y_n + h(A \otimes I_N)F(Y) + h^2(\widehat{A} \otimes I_N)G(Y),$$
$$y_{n+1} = y_n + h(b^T \otimes I_N)F(Y) + h^2(\widehat{b}^T \otimes I_N)G(Y),$$

where $e = [1]_{s\times 1}$, $A = [a_{ij}]_{s\times s}$, $\widehat{A} = [\widehat{a}_{ij}]_{s\times s}$, $b = [b_i]_{s\times 1}$, $\widehat{b} = [\widehat{b}_i]_{s\times 1}$, and

$$Y = \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_s \end{bmatrix}, \quad F(Y) = \begin{bmatrix} f(Y_1) \\ f(Y_2) \\ \vdots \\ f(Y_s) \end{bmatrix}, \quad G(Y) = \begin{bmatrix} g(Y_1) \\ g(Y_2) \\ \vdots \\ g(Y_s) \end{bmatrix}.$$

Basic Background

- Extended Butcher Tableau:

$$\begin{array}{c|c||c} c & A & \widehat{A} \\ \hline & b^T & \widehat{b}^T \end{array}$$

- Stability Function: For the standard test problem $y'(t) = \lambda y$, $y_{n+1} = R(z) y_n$, where

$$R(z) = 1 + (zb^T + z^2 \widehat{b}^T)(I - zA - z^2 \widehat{A})^{-1} e, \quad \text{with } z = h\lambda.$$

- Symmetry Conditions:

Basic Background

- Extended Butcher Tableau:

$$\begin{array}{c|c||c} c & A & \widehat{A} \\ \hline & b^T & \widehat{b}^T \end{array}$$

- Stability Function: For the standard test problem $y'(t) = \lambda y$, $y_{n+1} = R(z) y_n$, where

$$R(z) = 1 + (z b^T + z^2 \widehat{b}^T)(I - zA - z^2 \widehat{A})^{-1} e, \quad \text{with } z = h\lambda.$$

- Symmetry Conditions:

Basic Background

- Extended Butcher Tableau:

$$\begin{array}{c|c||c} c & A & \widehat{A} \\ \hline & b^T & \widehat{b}^T \end{array}$$

- Stability Function: For the standard test problem $y'(t) = \lambda y$, $y_{n+1} = R(z)y_n$, where

$$R(z) = 1 + (zb^T + z^2\widehat{b}^T)(I - zA - z^2\widehat{A})^{-1}e, \quad \text{with } z = h\lambda.$$

- Symmetry Conditions:

Basic Background

- Extended Butcher Tableau:

$$\begin{array}{c|c||c} c & A & \widehat{A} \\ \hline & b^T & \widehat{b}^T \end{array}$$

- Stability Function: For the standard test problem $y'(t) = \lambda y$, $y_{n+1} = R(z) y_n$, where

$$R(z) = 1 + (zb^T + z^2 \widehat{b}^T)(I - zA - z^2 \widehat{A})^{-1} e, \quad \text{with } z = h\lambda.$$

- Symmetry Conditions:

(t_n, y_n) •

Basic Background

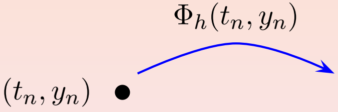
- Extended Butcher Tableau:

c	A	$\widehat{A}$
	b^T	$\widehat{b}^T$

- Stability Function: For the standard test problem $y'(t) = \lambda y$, $y_{n+1} = R(z)y_n$, where

$$R(z) = 1 + (zb^T + z^2\,\widehat{b}^T)(I - zA - z^2\widehat{A})^{-1}e, \quad \text{with } z = h\lambda.$$

- Symmetry Conditions:



Basic Background

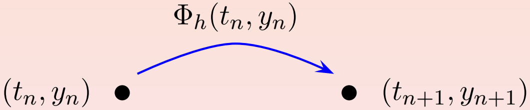
- Extended Butcher Tableau:

$$\begin{array}{c|c||c} c & A & \widehat{A} \\ \hline & b^T & \widehat{b}^T \end{array}$$

- Stability Function: For the standard test problem $y'(t) = \lambda y$, $y_{n+1} = R(z) y_n$, where

$$R(z) = 1 + (zb^T + z^2 \widehat{b}^T)(I - zA - z^2 \widehat{A})^{-1} e, \quad \text{with } z = h\lambda.$$

- Symmetry Conditions:



Basic Background

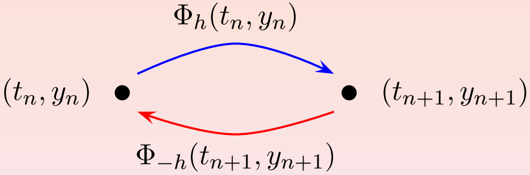
- Extended Butcher Tableau:

$$\begin{array}{c|c||c} c & A & \widehat{A} \\ \hline & b^T & \widehat{b}^T \end{array}$$

- Stability Function: For the standard test problem $y'(t) = \lambda y$, $y_{n+1} = R(z) y_n$, where

$$R(z) = 1 + (zb^T + z^2 \widehat{b}^T)(I - zA - z^2 \widehat{A})^{-1} e, \quad \text{with } z = h\lambda.$$

- Symmetry Conditions:



Basic Background

- Symmetry Conditions:

Basic Background

- Symmetry Conditions:

$$
\begin{aligned}
PAP &= eb^T - A \\
P\widehat{A}P &= -e\widehat{b}^T + \widehat{A} \\
Pb &= b \\
P\widehat{b} &= -\widehat{b}
\end{aligned}
$$

  where P is the permutation matrix which reverses the stages.

- Order Conditions: As for RK methods, we compare the Taylor Series expansions of the exact and numerical solutions, $y(t_n + h)$ and y_{n+1} respectively, to derive the order conditions of methods.

Basic Background

- Symmetry Conditions:

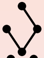
$$PAP = eb^T - A$$
$$P\widehat{A}P = -e\widehat{b}^T + \widehat{A}$$
$$Pb = b$$
$$P\widehat{b} = -\widehat{b}$$

  where P is the permutation matrix which reverses the stages.

- Order Conditions: As for RK methods, we compare the Taylor Series expansions of the exact and numerical solutions, $y(t_n + h)$ and y_{n+1} respectively, to derive the order conditions of methods.

Order Conditions

- Order conditions assuming $C(1)$:

Order	Tree	Order Condition
1		$b^T e = 1$
2		$b^T c + \widehat{b}^T e = \frac{1}{2}$
3		$b^T c^2 + 2\widehat{b}^T c = \frac{1}{3}$
		$b^T A c + b^T \widehat{A} e + \widehat{b}^T c = \frac{1}{6}$
4		$b^T c^3 + 3\widehat{b}^T c^2 = \frac{1}{4}$
		$b^T c A c + b^T c \widehat{A} e + \widehat{b}^T c^2 + \widehat{b}^T A c + \widehat{b}^T \widehat{A} e = \frac{1}{8}$
		$b^T A c^2 + 2 b^T \widehat{A} c + \widehat{b}^T c^2 = \frac{1}{12}$
		$b^T A^2 c + b^T A \widehat{A} e + b^T \widehat{A} c + \widehat{b}^T A c + \widehat{b}^T \widehat{A} e = \frac{1}{24}$

Order Conditions

- Order conditions assuming $C(1)$:

Order	Tree	Order Condition
1		$b^T e = 1$
2		$b^T c + \widehat{b}^T e = \frac{1}{2}$
3		$b^T c^2 + 2\widehat{b}^T c = \frac{1}{3}$
		$b^T Ac + b^T \widehat{A} e + \widehat{b}^T c = \frac{1}{6}$
4		$b^T c^3 + 3\widehat{b}^T c^2 = \frac{1}{4}$
		$b^T cAc + b^T c\widehat{A}e + \widehat{b}^T c^2 + \widehat{b}^T Ac + \widehat{b}^T \widehat{A}e = \frac{1}{8}$
		$b^T Ac^2 + 2b^T \widehat{A}c + \widehat{b}^T c^2 = \frac{1}{12}$
		$b^T A^2 c + b^T A\widehat{A}e + b^T \widehat{A}c + \widehat{b}^T Ac + \widehat{b}^T \widehat{A}e = \frac{1}{24}$

Simplifying Assumptions and Labelling Trees

- Stage Order Conditions:

$$C(q): \quad Ac^{k-1} + (k-1)\widehat{A}c^{k-2} = \frac{c^k}{k}, \quad k = 1, \ldots, q.$$

- Bushy Tree Conditions:
- Example of Labelling Trees:

Simplifying Assumptions and Labelling Trees

- Stage Order Conditions:

$$C(q): \quad Ac^{k-1} + (k-1)\widehat{A}c^{k-2} = \frac{c^k}{k}, \quad k = 1, \ldots, q.$$

- Bushy Tree Conditions:

$$B(p): \quad b^T c^{k-1} + (k-1)\widehat{b}^T c^{k-2} = \frac{1}{k}, \quad k = 1, \ldots, p.$$

- Example of Labelling Trees:

Simplifying Assumptions and Labelling Trees

- Stage Order Conditions:

$$C(q): \quad Ac^{k-1} + (k-1)\widehat{A}c^{k-2} = \frac{c^k}{k}, \quad k = 1, \ldots, q.$$

- Bushy Tree Conditions: 

$$B(p): \quad b^T c^{k-1} + (k-1)\widehat{b}^T c^{k-2} = \frac{1}{k}, \quad k = 1, \ldots, p.$$

- Example of Labelling Trees:

Simplifying Assumptions and Labelling Trees

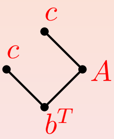
- Stage Order Conditions:

$$C(q): \quad Ac^{k-1} + (k-1)\widehat{A}c^{k-2} = \frac{c^k}{k}, \quad k = 1, \ldots, q.$$

- Bushy Tree Conditions: 

$$B(p): \quad b^T c^{k-1} + (k-1)\widehat{b}^T c^{k-2} = \frac{1}{k}, \quad k = 1, \ldots, p.$$

- Example of Labelling Trees:



Simplifying Assumptions and Labelling Trees

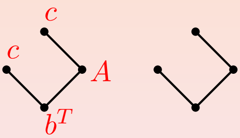
- Stage Order Conditions:

$$C(q): \quad Ac^{k-1} + (k-1)\widehat{A}c^{k-2} = \frac{c^k}{k}, \quad k = 1, \ldots, q.$$

- Bushy Tree Conditions: 

$$B(p): \quad b^T c^{k-1} + (k-1)\widehat{b}^T c^{k-2} = \frac{1}{k}, \quad k = 1, \ldots, p.$$

- Example of Labelling Trees:



Simplifying Assumptions and Labelling Trees

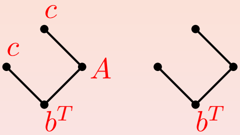
- Stage Order Conditions:

$$C(q): \quad Ac^{k-1} + (k-1)\widehat{A}c^{k-2} = \frac{c^k}{k}, \quad k = 1, \ldots, q.$$

- Bushy Tree Conditions: 

$$B(p): \quad b^T c^{k-1} + (k-1)\widehat{b}^T c^{k-2} = \frac{1}{k}, \quad k = 1, \ldots, p.$$

- Example of Labelling Trees:



Simplifying Assumptions and Labelling Trees

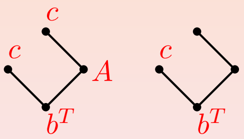
- Stage Order Conditions:

$$C(q): \quad Ac^{k-1} + (k-1)\widehat{A}c^{k-2} = \frac{c^k}{k}, \quad k = 1, \ldots, q.$$

- Bushy Tree Conditions: 

$$B(p): \quad b^T c^{k-1} + (k-1)\widehat{b}^T c^{k-2} = \frac{1}{k}, \quad k = 1, \ldots, p.$$

- Example of Labelling Trees:



Simplifying Assumptions and Labelling Trees

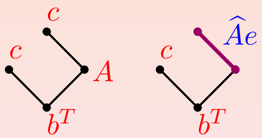
- Stage Order Conditions:

$$C(q): \quad Ac^{k-1} + (k-1)\widehat{A}c^{k-2} = \frac{c^k}{k}, \quad k = 1, \dots, q.$$

- Bushy Tree Conditions:

$$B(p): \quad b^T c^{k-1} + (k-1)\widehat{b}^T c^{k-2} = \frac{1}{k}, \quad k = 1, \dots, p.$$

- Example of Labelling Trees:



Simplifying Assumptions and Labelling Trees

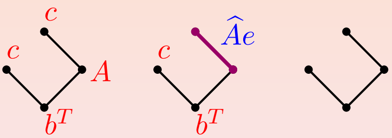
- Stage Order Conditions:

$$C(q): \quad Ac^{k-1} + (k-1)\widehat{A}c^{k-2} = \frac{c^k}{k}, \quad k = 1, \ldots, q.$$

- Bushy Tree Conditions: 

$$B(p): \quad b^T c^{k-1} + (k-1)\widehat{b}^T c^{k-2} = \frac{1}{k}, \quad k = 1, \ldots, p.$$

- Example of Labelling Trees:



Simplifying Assumptions and Labelling Trees

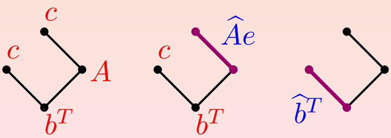
- Stage Order Conditions:

$$C(q): \quad Ac^{k-1} + (k-1)\widehat{A}c^{k-2} = \frac{c^k}{k}, \quad k = 1, \dots, q.$$

- Bushy Tree Conditions: 

$$B(p): \quad b^T c^{k-1} + (k-1)\widehat{b}^T c^{k-2} = \frac{1}{k}, \quad k = 1, \dots, p.$$

- Example of Labelling Trees:



Simplifying Assumptions and Labelling Trees

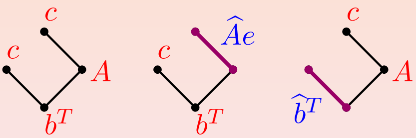
- Stage Order Conditions:

$$C(q): \quad Ac^{k-1} + (k-1)\widehat{A}c^{k-2} = \frac{c^k}{k}, \quad k = 1, \dots, q.$$

- Bushy Tree Conditions: 

$$B(p): \quad b^T c^{k-1} + (k-1)\widehat{b}^T c^{k-2} = \frac{1}{k}, \quad k = 1, \dots, p.$$

- Example of Labelling Trees:



Simplifying Assumptions and Labelling Trees

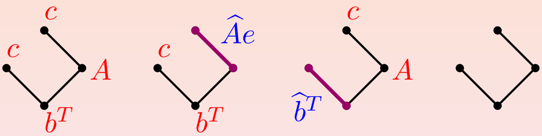
- Stage Order Conditions:

$$C(q): \quad Ac^{k-1} + (k-1)\widehat{A}c^{k-2} = \frac{c^k}{k}, \quad k = 1, \ldots, q.$$

- Bushy Tree Conditions: 

$$B(p): \quad b^T c^{k-1} + (k-1)\widehat{b}^T c^{k-2} = \frac{1}{k}, \quad k = 1, \ldots, p.$$

- Example of Labelling Trees:



Simplifying Assumptions and Labelling Trees

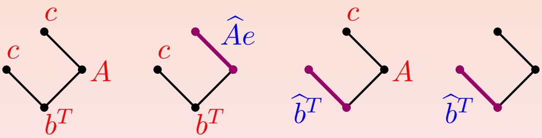
- Stage Order Conditions:

$$C(q): \quad Ac^{k-1} + (k-1)\widehat{A}c^{k-2} = \frac{c^k}{k}, \quad k = 1, \ldots, q.$$

- Bushy Tree Conditions: 

$$B(p): \quad b^T c^{k-1} + (k-1)\widehat{b}^T c^{k-2} = \frac{1}{k}, \quad k = 1, \ldots, p.$$

- Example of Labelling Trees:



Simplifying Assumptions and Labelling Trees

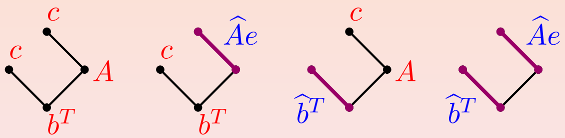
- Stage Order Conditions:

$$C(q): \quad Ac^{k-1} + (k-1)\widehat{A}c^{k-2} = \frac{c^k}{k}, \quad k = 1, \dots, q.$$

- Bushy Tree Conditions: 

$$B(p): \quad b^T c^{k-1} + (k-1)\widehat{b}^T c^{k-2} = \frac{1}{k}, \quad k = 1, \dots, p.$$

- Example of Labelling Trees:



Simplifying Assumptions and Labelling Trees

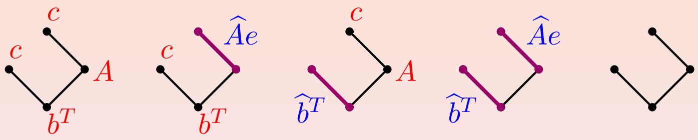
- Stage Order Conditions:

$$C(q): \quad Ac^{k-1} + (k-1)\widehat{A}c^{k-2} = \frac{c^k}{k}, \quad k = 1, \ldots, q.$$

- Bushy Tree Conditions:

$$B(p): \quad b^T c^{k-1} + (k-1)\widehat{b}^T c^{k-2} = \frac{1}{k}, \quad k = 1, \ldots, p.$$

- Example of Labelling Trees:



Simplifying Assumptions and Labelling Trees

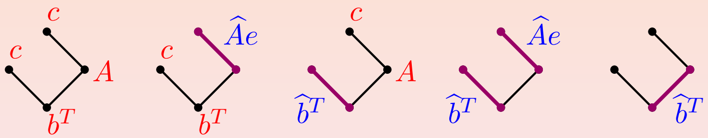
- Stage Order Conditions:

$$C(q): \quad Ac^{k-1} + (k-1)\widehat{A}c^{k-2} = \frac{c^k}{k}, \quad k = 1, \ldots, q.$$

- Bushy Tree Conditions: 

$$B(p): \quad b^T c^{k-1} + (k-1)\widehat{b}^T c^{k-2} = \frac{1}{k}, \quad k = 1, \ldots, p.$$

- Example of Labelling Trees:



Simplifying Assumptions and Labelling Trees

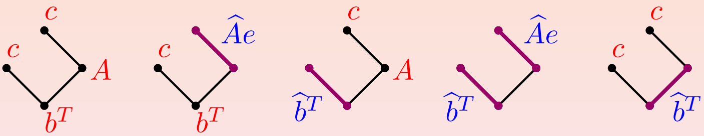
- Stage Order Conditions:

$$C(q): \quad Ac^{k-1} + (k-1)\widehat{A}c^{k-2} = \frac{c^k}{k}, \quad k = 1, \ldots, q.$$

- Bushy Tree Conditions: 

$$B(p): \quad b^T c^{k-1} + (k-1)\widehat{b}^T c^{k-2} = \frac{1}{k}, \quad k = 1, \ldots, p.$$

- Example of Labelling Trees:



Simplifying Assumptions and Labelling Trees

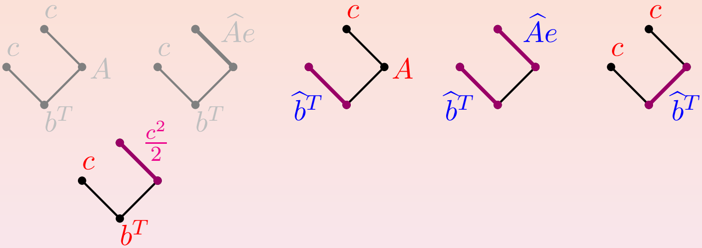
- Stage Order Conditions:

$$C(q): \quad Ac^{k-1} + (k-1)\widehat{A}c^{k-2} = \frac{c^k}{k}, \quad k = 1, \ldots, q.$$

- Bushy Tree Conditions: 

$$B(p): \quad b^T c^{k-1} + (k-1)\widehat{b}^T c^{k-2} = \frac{1}{k}, \quad k = 1, \ldots, p.$$

- Example of Labelling Trees:



Simplifying Assumptions and Labelling Trees

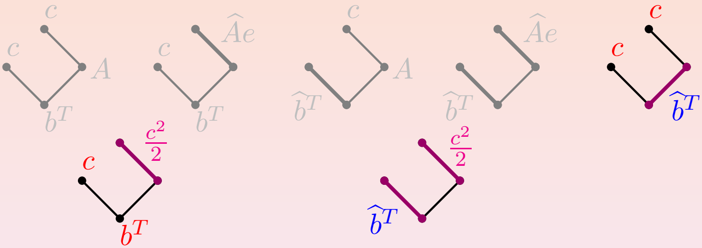
- Stage Order Conditions:

$$C(q): \quad Ac^{k-1} + (k-1)\widehat{A}c^{k-2} = \frac{c^k}{k}, \quad k = 1, \ldots, q.$$

- Bushy Tree Conditions: 

$$B(p): \quad b^T c^{k-1} + (k-1)\widehat{b}^T c^{k-2} = \frac{1}{k}, \quad k = 1, \ldots, p.$$

- Example of Labelling Trees:



Simplifying Assumptions and Labelling Trees

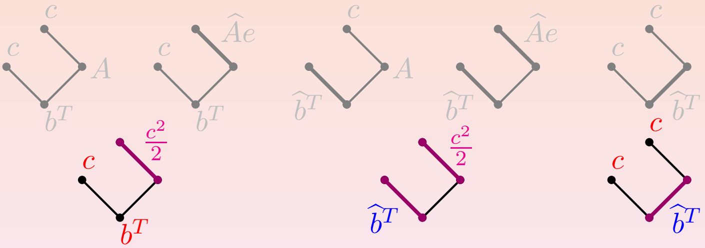
- Stage Order Conditions:

$$C(q): \quad Ac^{k-1} + (k-1)\widehat{A}c^{k-2} = \frac{c^k}{k}, \quad k = 1, \dots, q.$$

- Bushy Tree Conditions: 

$$B(p): \quad b^T c^{k-1} + (k-1)\widehat{b}^T c^{k-2} = \frac{1}{k}, \quad k = 1, \dots, p.$$

- Example of Labelling Trees:



Simplifying Assumptions and Labelling Trees

- Stage Order Conditions:

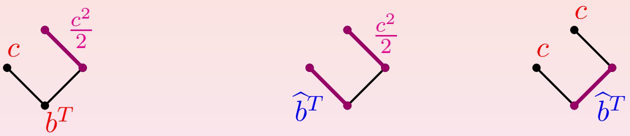
$$C(q): \quad Ac^{k-1} + (k-1)\widehat{A}c^{k-2} = \frac{c^k}{k}, \quad k = 1, \dots, q.$$

- Bushy Tree Conditions: 

$$B(p): \quad b^T c^{k-1} + (k-1)\widehat{b}^T c^{k-2} = \frac{1}{k}, \quad k = 1, \dots, p.$$

- Example of Labelling Trees:

$$\tfrac{1}{2} b^T c^3 + \tfrac{1}{2}\widehat{b}^T c^2 + \widehat{b}^T c^2 = \tfrac{1}{8}$$



Simplifying Assumptions and Labelling Trees

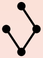
- Stage Order Conditions:

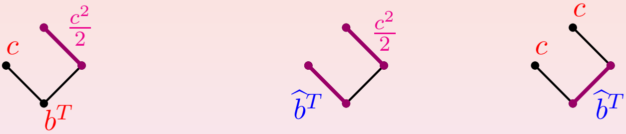
$$C(q): \quad Ac^{k-1} + (k-1)\widehat{A}c^{k-2} = \frac{c^k}{k}, \quad k = 1, \ldots, q.$$

- Bushy Tree Conditions: 

$$B(p): \quad b^T c^{k-1} + (k-1)\widehat{b}^T c^{k-2} = \frac{1}{k}, \quad k = 1, \ldots, p.$$

- Example of Labelling Trees:

$$\tfrac{1}{2}b^T c^3 + \tfrac{3}{2}\widehat{b}^T c^2 = \tfrac{1}{8}$$ 



Simplifying Assumptions and Labelling Trees

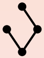
- Stage Order Conditions:

$$C(q): \quad Ac^{k-1} + (k-1)\widehat{A}c^{k-2} = \frac{c^k}{k}, \quad k = 1, \ldots, q.$$

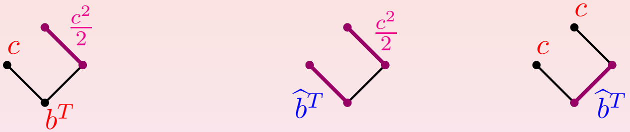
- Bushy Tree Conditions: 

$$B(p): \quad b^T c^{k-1} + (k-1)\widehat{b}^T c^{k-2} = \frac{1}{k}, \quad k = 1, \ldots, p.$$

- Example of Labelling Trees:

$$\tfrac{1}{2} b^T c^3 + \tfrac{3}{2}\widehat{b}^T c^2 = \tfrac{1}{8}$$ 

$$b^T c^3 + 3\widehat{b}^T c^2 = \tfrac{1}{4}$$ 



Constructing Explicit TDRK Methods

- In our study, we include two special groups of explicit TDRK methods:

GROUP A:

$$\begin{array}{c|c||c} c & Ae_1 & \widehat{A} \\ \hline & b_1 & \widehat{b}^T \end{array} \Rightarrow \begin{array}{c||c} c & \widehat{A} \\ \hline & \widehat{b}^T \end{array}$$

GROUP B:

$$\begin{array}{c|c||c} c & A & \widehat{A}e_1 \\ \hline & b^T & \widehat{b}_1 \end{array}$$

- We also constructed embedded explicit TDRK methods to compare with some popular embedded explicit RK methods.
- Explicit TDRK methods can easily have stage order 2, i.e. they satisfy the $C(2)$ conditions.

Constructing Explicit TDRK Methods

- In our study, we include two special groups of explicit TDRK methods:

GROUP A:

$$\begin{array}{c|c||c} c & Ae_1 & \widehat{A} \\ \hline & b_1 & \widehat{b}^T \end{array} \Rightarrow \begin{array}{c||c} c & \widehat{A} \\ \hline & \widehat{b}^T \end{array}$$

GROUP B:

$$\begin{array}{c|c||c} c & A & \widehat{A}e_1 \\ \hline & b^T & \widehat{b}_1 \end{array}$$

- We also constructed embedded explicit TDRK methods to compare with some popular embedded explicit RK methods.
- Explicit TDRK methods can easily have stage order 2, i.e. they satisfy the $C(2)$ conditions.

Constructing Explicit TDRK Methods

- In our study, we include two special groups of explicit TDRK methods:

GROUP A:

$$\begin{array}{c|c||c} c & Ae_1 & \widehat{A} \\ \hline & b_1 & \widehat{b}^T \end{array} \Rightarrow \begin{array}{c||c} c & \widehat{A} \\ \hline & \widehat{b}^T \end{array}$$

GROUP B:

$$\begin{array}{c|c||c} c & A & \widehat{A}e_1 \\ \hline & b^T & \widehat{b}_1 \end{array}$$

- We also constructed embedded explicit TDRK methods to compare with some popular embedded explicit RK methods.
- Explicit TDRK methods can easily have stage order 2, i.e. they satisfy the $C(2)$ conditions.

Constructing Explicit TDRK Methods

- In our study, we include two special groups of explicit TDRK methods:

GROUP A:

$$\begin{array}{c|c||c} c & Ae_1 & \widehat{A} \\ \hline & b_1 & \widehat{b}^T \end{array} \Rightarrow \begin{array}{c||c} c & \widehat{A} \\ \hline & \widehat{b}^T \end{array}$$

GROUP B:

$$\begin{array}{c|c||c} c & A & \widehat{A}e_1 \\ \hline & b^T & \widehat{b}_1 \end{array}$$

- We also constructed embedded explicit TDRK methods to compare with some popular embedded explicit RK methods.
- Explicit TDRK methods can easily have stage order 2, i.e. they satisfy the $C(2)$ conditions.

Constructing Explicit TDRK Methods

- In our study, we include two special groups of explicit TDRK methods:

GROUP A:

$$\begin{array}{c|c||c} c & Ae_1 & \widehat{A} \\ \hline & b_1 & \widehat{b}^T \end{array} \Rightarrow \begin{array}{c||c} c & \widehat{A} \\ \hline & \widehat{b}^T \end{array}$$

GROUP B:

$$\begin{array}{c|c||c} c & A & \widehat{A}e_1 \\ \hline & b^T & \widehat{b}_1 \end{array}$$

- We also constructed embedded explicit TDRK methods to compare with some popular embedded explicit RK methods.
- Explicit TDRK methods can easily have stage order 2, i.e. they satisfy the $C(2)$ conditions.

Constructing Explicit TDRK Methods

- In our study, we include two special groups of explicit TDRK methods:

GROUP A:

$$\begin{array}{c|c||c} c & Ae_1 & \widehat{A} \\ \hline & b_1 & \widehat{b}^T \end{array} \Rightarrow \begin{array}{c||c} c & \widehat{A} \\ \hline & \widehat{b}^T \end{array}$$

GROUP B:

$$\begin{array}{c|c||c} c & A & \widehat{A}e_1 \\ \hline & b^T & \widehat{b}_1 \end{array}$$

- We also constructed embedded explicit TDRK methods to compare with some popular embedded explicit RK methods.
- Explicit TDRK methods can easily have stage order 2, i.e. they satisfy the $C(2)$ conditions.

Constructing Explicit TDRK Methods

- TDRK45b/TDRK5b: $p = 5, q = 2$

0	0	0	0	0	
$\frac{1}{3}$	$\frac{1}{18}$	0	0	0	
$\frac{1}{2}$	$\frac{1}{8}$	0	0	0	
$\frac{4}{5}$	$-\frac{2}{125}$	$\frac{42}{125}$	0	0	
1	$\frac{5}{48}$	$\frac{9}{28}$	0	$\frac{25}{336}$	order 5
1	$\frac{1}{6}$	0	$\frac{1}{3}$	0	order 4

- TDRK5b requires $1f + 3g$ function evaluations per step, and $R(z) = 1 + z + \frac{z^2}{2} + \frac{z^3}{6} + \frac{z^4}{24} + \frac{z^5}{120} + \frac{z^6}{720}$.
- TDRK45b is an embedded method which requires $1f + 4g$ function evaluations per step.

Constructing Explicit TDRK Methods

- TDRK45b/TDRK5b: $p = 5, q = 2$

$$
\begin{array}{c||cccc l}
0 & 0 & 0 & 0 & 0 & \\
\frac{1}{3} & \frac{1}{18} & 0 & 0 & 0 & \\
\frac{1}{2} & \frac{1}{8} & 0 & 0 & 0 & \\
\frac{4}{5} & -\frac{2}{125} & \frac{42}{125} & 0 & 0 & \\
\hline
1 & \frac{5}{48} & \frac{9}{28} & 0 & \frac{25}{336} & \text{order 5} \\
\hline
1 & \frac{1}{6} & 0 & \frac{1}{3} & 0 & \text{order 4}
\end{array}
$$

- TDRK5b requires $1f + 3g$ function evaluations per step, and $R(z) = 1 + z + \frac{z^2}{2} + \frac{z^3}{6} + \frac{z^4}{24} + \frac{z^5}{120} + \frac{z^6}{720}$.
- TDRK45b is an embedded method which requires $1f + 4g$ function evaluations per step.

Constructing Explicit TDRK Methods

- TDRK45b/TDRK5b: $p = 5, q = 2$

$$
\begin{array}{c||cccc}
0 & 0 & 0 & 0 & 0 \\
\frac{1}{3} & \frac{1}{18} & 0 & 0 & 0 \\
\frac{1}{2} & \frac{1}{8} & 0 & 0 & 0 \\
\frac{4}{5} & -\frac{2}{125} & \frac{42}{125} & 0 & 0 \\
\hline
1 & \frac{5}{48} & \frac{9}{28} & 0 & \frac{25}{336} \\
\hline
1 & \frac{1}{6} & 0 & \frac{1}{3} & 0
\end{array}
\quad
\begin{array}{l}
\\ \\ \\ \\
\text{order 5} \\
\text{order 4}
\end{array}
$$

- TDRK5b requires $1f + 3g$ function evaluations per step, and $R(z) = 1 + z + \frac{z^2}{2} + \frac{z^3}{6} + \frac{z^4}{24} + \frac{z^5}{120} + \frac{z^6}{720}$.
- TDRK45b is an embedded method which requires $1f + 4g$ function evaluations per step.

Constructing Implicit TDRK Methods

- We have constructed several implicit TDRK methods, for example, TDRK244sss is a 2-stage, order-4, stage-order-4, semi-implicit, symmetric, and stiffly-accurate method:

$$\begin{array}{c|cc||cc} 0 & 0 & 0 & 0 & 0 \\ 1 & \frac{1}{2} & \frac{1}{2} & \frac{1}{12} & -\frac{1}{12} \\ \hline & \frac{1}{2} & \frac{1}{2} & \frac{1}{12} & -\frac{1}{12} \end{array} \qquad R(z) = \frac{12 + 6z + z^2}{12 - 6z + z^2}$$

- The implicit TDRK methods we constructed range from order 3 to 6, the order-3 and 5 methods are L-stable and the order-4 and 6 methods are A-stable.

Constructing Implicit TDRK Methods

- We have constructed several implicit TDRK methods, for example, TDRK244sss is a 2-stage, order-4, stage-order-4, semi-implicit, symmetric, and stiffly-accurate method:

$$
\begin{array}{c|cc||cc}
0 & 0 & 0 & 0 & 0 \\
1 & \frac{1}{2} & \frac{1}{2} & \frac{1}{12} & -\frac{1}{12} \\
\hline
 & \frac{1}{2} & \frac{1}{2} & \frac{1}{12} & -\frac{1}{12}
\end{array}
\qquad
R(z) = \frac{12 + 6z + z^2}{12 - 6z + z^2}
$$

- The implicit TDRK methods we constructed range from order 3 to 6, the order-3 and 5 methods are L-stable and the order-4 and 6 methods are A-stable.

Constructing Implicit TDRK Methods

- We have constructed several implicit TDRK methods, for example, TDRK244sss is a 2-stage, order-4, stage-order-4, semi-implicit, symmetric, and stiffly-accurate method:

$$\begin{array}{c|cc||cc} 0 & 0 & 0 & 0 & 0 \\ 1 & \frac{1}{2} & \frac{1}{2} & \frac{1}{12} & -\frac{1}{12} \\ \hline & \frac{1}{2} & \frac{1}{2} & \frac{1}{12} & -\frac{1}{12} \end{array} \qquad R(z) = \frac{12 + 6z + z^2}{12 - 6z + z^2}$$

- The implicit TDRK methods we constructed range from order 3 to 6, the order-3 and 5 methods are L-stable and the order-4 and 6 methods are A-stable.

Stiff ODE Problems

- Prothero-Robinson Problem (PR):

$$y'(t) = \lambda(y(t) - \phi(t)) + \phi'(t),$$

we show the results for $\phi(t) = \sin(t)$ and two cases for the implicit methods,

  - PR1b: $y_0 = \phi_0$ and $\lambda = -10^4$. Exact solution is $y(t) = \phi(t)$.
  - PR1d: $y_0 = 1$ and $\lambda = -10^4$. Exact solution is $y(t) = \phi(t) + (y_0 - \phi_0)\exp(\lambda t)$.

- Kaps Problem:

$$y'(t) = \begin{bmatrix} -y_1(1+y_1) + y_2 \\ \lambda(y_1^2 - y_2) - 2y_2 \end{bmatrix}, \quad y(0) = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad \text{Re}(\lambda) \gg 1,$$

with exact solution $y(t) = [\exp(-t), \exp(-2t)]^T$.

Stiff ODE Problems

- Prothero-Robinson Problem (PR):

$$y'(t) = \lambda(y(t) - \phi(t)) + \phi'(t),$$

we show the results for $\phi(t) = \sin(t)$ and two cases for the implicit methods,

  - PR1b: $y_0 = \phi_0$ and $\lambda = -10^4$. Exact solution is $y(t) = \phi(t)$.
  - PR1d: $y_0 = 1$ and $\lambda = -10^4$. Exact solution is $y(t) = \phi(t) + (y_0 - \phi_0)\exp(\lambda t)$.

- Kaps Problem:

$$y'(t) = \begin{bmatrix} -y_1(1+y_1) + y_2 \\ \lambda(y_1^2 - y_2) - 2y_2 \end{bmatrix}, \quad y(0) = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad \mathrm{Re}(\lambda) \gg 1,$$

with exact solution $y(t) = [\exp(-t), \exp(-2t)]^T$.

Stiff ODE Problems

- Prothero-Robinson Problem (PR):

$$y'(t) = \lambda(y(t) - \phi(t)) + \phi'(t),$$

we show the results for $\phi(t) = \sin(t)$ and two cases for the implicit methods,

  - PR1b: $y_0 = \phi_0$ and $\lambda = -10^4$. Exact solution is $y(t) = \phi(t)$.
  - PR1d: $y_0 = 1$ and $\lambda = -10^4$. Exact solution is $y(t) = \phi(t) + (y_0 - \phi_0)\exp(\lambda t)$.

- Kaps Problem:

$$y'(t) = \begin{bmatrix} -y_1(1+y_1) + y_2 \\ \lambda(y_1^2 - y_2) - 2y_2 \end{bmatrix}, \quad y(0) = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad \text{Re}(\lambda) \gg 1,$$

with exact solution $y(t) = [\exp(-t), \exp(-2t)]^T$.

Stiff ODE Problems

- Prothero-Robinson Problem (PR):

$$y'(t) = \lambda(y(t) - \phi(t)) + \phi'(t),$$

we show the results for $\phi(t) = \sin(t)$ and two cases for the implicit methods,

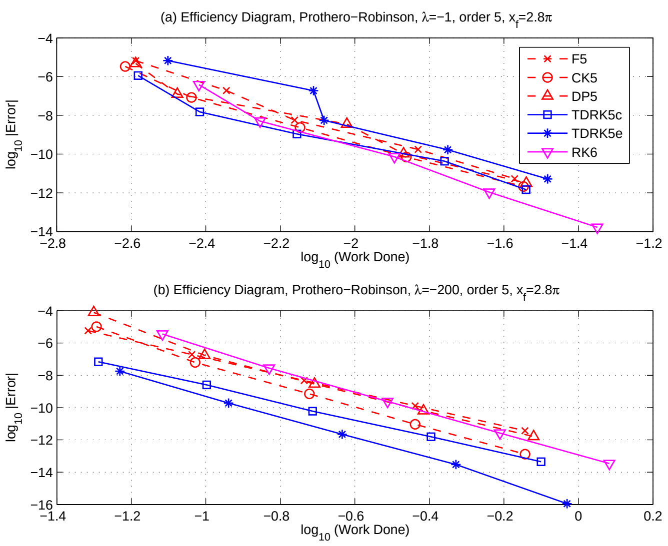
  - PR1b: $y_0 = \phi_0$ and $\lambda = -10^4$. Exact solution is $y(t) = \phi(t)$.
  - PR1d: $y_0 = 1$ and $\lambda = -10^4$. Exact solution is $y(t) = \phi(t) + (y_0 - \phi_0)\exp(\lambda t)$.

- Kaps Problem:

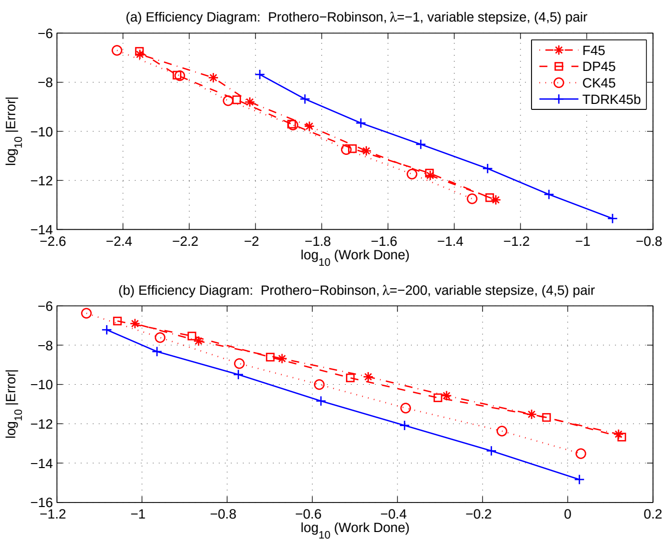
$$y'(t) = \begin{bmatrix} -y_1(1+y_1) + y_2 \\ \lambda(y_1^2 - y_2) - 2y_2 \end{bmatrix}, \quad y(0) = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad \text{Re}(\lambda) \gg 1,$$

with exact solution $y(t) = [\exp(-t), \exp(-2t)]^T$.

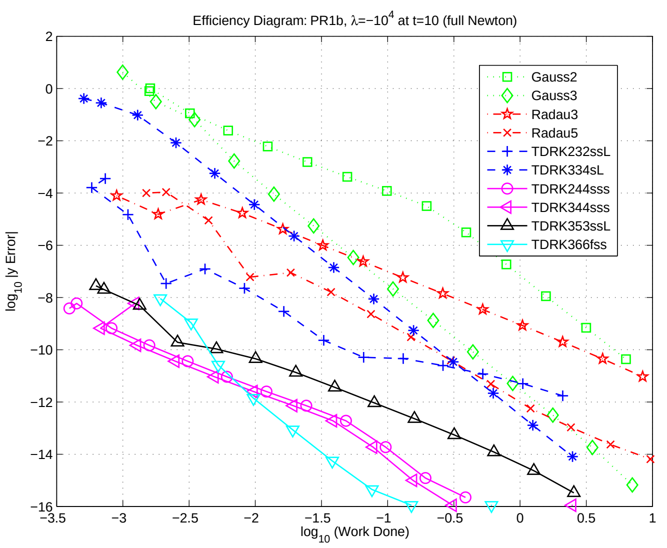
Explicit Methods for PR Problem



Embedded Explicit Methods for PR Problem



Implicit Methods for PR Problem



Implicit Methods for PR Problem

- Order Behaviour – Error for PR problem when $h \to 0$ and $z = \lambda h \to \infty$:

Implicit Methods for PR Problem

- Order Behaviour – Error for PR problem when $h \to 0$ and $z = \lambda h \to \infty$:

p	Method	Local Error	Global Error

Implicit Methods for PR Problem

- Order Behaviour – Error for PR problem when $h \to 0$ and $z = \lambda h \to \infty$:

p	Method	Local Error	Global Error
3	TDRK232ssL	$\begin{cases} O(h^3/z^2) = O(h/\lambda^2) \\ O(h^4/z) = O(h^3/\lambda) \end{cases}$	$\begin{cases} O(h/\lambda^2) & \text{small } h \\ O(h^3/\lambda) & \text{large } h \end{cases}$

Implicit Methods for PR Problem

- Order Behaviour – Error for PR problem when $h \to 0$ and $z = \lambda h \to \infty$:

p	Method	Local Error	Global Error
3	TDRK232ssL	$\begin{cases} O(h^3/z^2) = O(h/\lambda^2) \\ O(h^4/z) = O(h^3/\lambda) \end{cases}$	$\begin{cases} O(h/\lambda^2) & \text{small } h \\ O(h^3/\lambda) & \text{large } h \end{cases}$
3	TDRK334sL	$O(h^4)$	$O(h^4)$

Implicit Methods for PR Problem

- Order Behaviour – Error for PR problem when $h \to 0$ and $z = \lambda h \to \infty$:

p	Method	Local Error	Global Error
3	TDRK232ssL	$\begin{cases} O(h^3/z^2) = O(h/\lambda^2) \\ O(h^4/z) = O(h^3/\lambda) \end{cases}$	$\begin{cases} O(h/\lambda^2) & \text{small } h \\ O(h^3/\lambda) & \text{large } h \end{cases}$
3	TDRK334sL	$O(h^4)$	$O(h^4)$
4	TDRK244sss	$O(h^5/z^2) = O(h^3/\lambda^2)$	$O(h^2/\lambda^2)$
4	TDRK344sss	$O(h^5/z^2) = O(h^3/\lambda^2)$	$O(h^2/\lambda^2)$

Implicit Methods for PR Problem

- Order Behaviour – Error for PR problem when $h \to 0$ and $z = \lambda h \to \infty$:

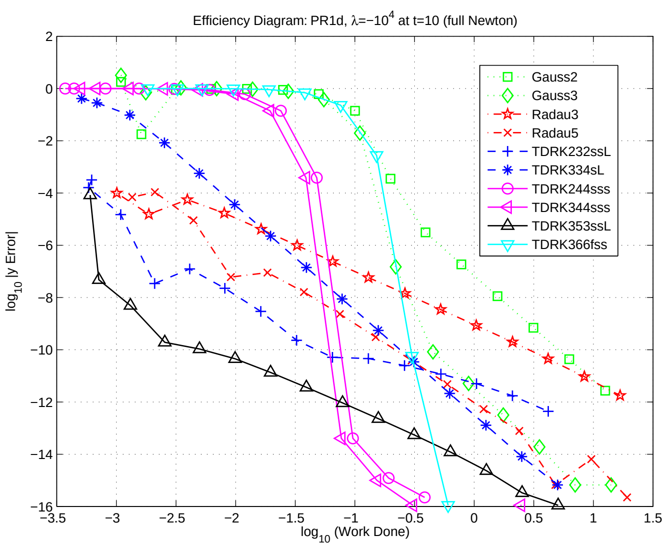
p	Method	Local Error	Global Error
3	TDRK232ssL	$\begin{cases} O(h^3/z^2) = O(h/\lambda^2) \\ O(h^4/z) = O(h^3/\lambda) \end{cases}$	$\begin{cases} O(h/\lambda^2) & \text{small } h \\ O(h^3/\lambda) & \text{large } h \end{cases}$
3	TDRK334sL	$O(h^4)$	$O(h^4)$
4	TDRK244sss	$O(h^5/z^2) = O(h^3/\lambda^2)$	$O(h^2/\lambda^2)$
4	TDRK344sss	$O(h^5/z^2) = O(h^3/\lambda^2)$	$O(h^2/\lambda^2)$
5	TDRK353ssL	$O(h^4/z^2) = O(h^2/\lambda^2)$	$O(h^2/\lambda^2)$

Implicit Methods for PR Problem

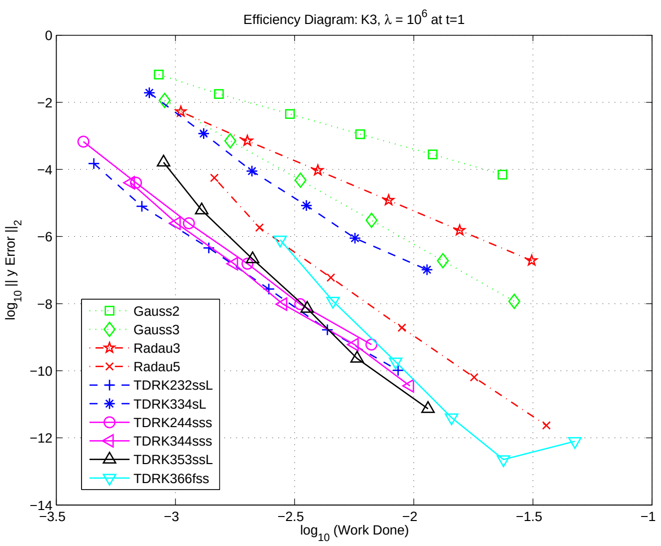
- Order Behaviour – Error for PR problem when $h \to 0$ and $z = \lambda h \to \infty$:

p	Method	Local Error	Global Error
3	TDRK232ssL	$\begin{cases} O(h^3/z^2) = O(h/\lambda^2) \\ O(h^4/z) = O(h^3/\lambda) \end{cases}$	$\begin{cases} O(h/\lambda^2) & \text{small } h \\ O(h^3/\lambda) & \text{large } h \end{cases}$
3	TDRK334sL	$O(h^4)$	$O(h^4)$
4	TDRK244sss	$O(h^5/z^2) = O(h^3/\lambda^2)$	$O(h^2/\lambda^2)$
4	TDRK344sss	$O(h^5/z^2) = O(h^3/\lambda^2)$	$O(h^2/\lambda^2)$
5	TDRK353ssL	$O(h^4/z^2) = O(h^2/\lambda^2)$	$O(h^2/\lambda^2)$
6	TDRK366fss	$O(h^7/z^2) = O(h^5/\lambda^2)$	$O(h^4/\lambda^2)$

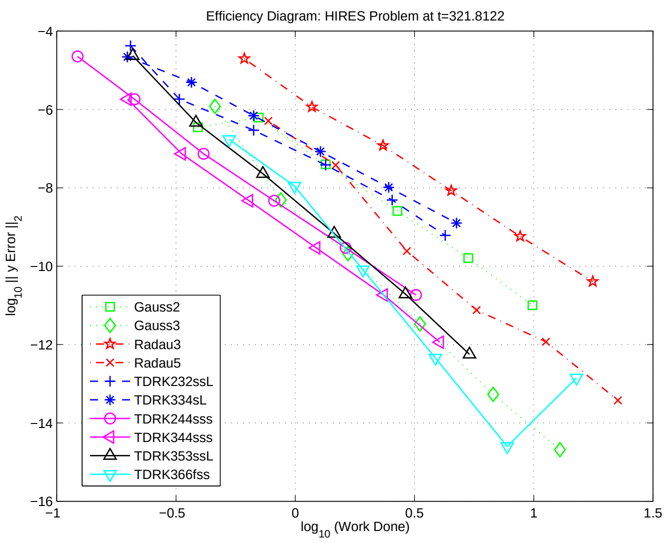
Implicit Methods for PR Problem



Implicit Methods for Kaps Problem



Implicit Methods for HIRES Problem



Classical PDE Methods

- Semi-discretization (or Method of Lines) is used to approximate PDEs by
  - firstly, discretize the spatial variables of PDEs to get a set of ODEs,
  - and then integrate along the time variable.
- However, many popular classical PDE methods are not MOL. Why?
- Two main disadvantages of MOL:
  - Stability is restricted by spatial discretization, possibly leading to unstable methods.
  - Approximation to higher order derivatives depends on the discretization used and often leads to non-optimal spread-out schemes.

Classical PDE Methods

- Semi-discretization (or Method of Lines) is used to approximate PDEs by
  - firstly, discretize the spatial variables of PDEs to get a set of ODEs,
  - and then integrate along the time variable.
- However, many popular classical PDE methods are not MOL. Why?
- Two main disadvantages of MOL:
  - Stability is restricted by spatial discretization, possibly leading to unstable methods.
  - Approximation to higher order derivatives depends on the discretization used and often leads to non-optimal spread-out schemes.

Classical PDE Methods

- Semi-discretization (or Method of Lines) is used to approximate PDEs by
  - firstly, discretize the spatial variables of PDEs to get a set of ODEs,
  - and then integrate along the time variable.
- However, many popular classical PDE methods are not MOL. Why?
- Two main disadvantages of MOL:
  - Stability is restricted by spatial discretization, possibly leading to unstable methods.
  - Approximation to higher order derivatives depends on the discretization used and often leads to non-optimal spread-out schemes.

Classical PDE Methods

- Semi-discretization (or Method of Lines) is used to approximate PDEs by
  - firstly, discretize the spatial variables of PDEs to get a set of ODEs,
  - and then integrate along the time variable.
- However, many popular classical PDE methods are not MOL. Why?
- Two main disadvantages of MOL:
  - Stability is restricted by spatial discretization, possibly leading to unstable methods.
  - Approximation to higher order derivatives depends on the discretization used and often leads to non-optimal spread-out schemes.

Classical PDE Methods

- Semi-discretization (or Method of Lines) is used to approximate PDEs by
  - firstly, discretize the spatial variables of PDEs to get a set of ODEs,
  - and then integrate along the time variable.
- However, many popular classical PDE methods are not MOL. Why?
- Two main disadvantages of MOL:
  - Stability is restricted by spatial discretization, possibly leading to unstable methods.
  - Approximation to higher order derivatives depends on the discretization used and often leads to non-optimal spread-out schemes.

Classical PDE Methods

- Semi-discretization (or Method of Lines) is used to approximate PDEs by
  - firstly, discretize the spatial variables of PDEs to get a set of ODEs,
  - and then integrate along the time variable.
- However, many popular classical PDE methods are not MOL. Why?
- Two main disadvantages of MOL:
  - Stability is restricted by spatial discretization, possibly leading to unstable methods.
  - Approximation to higher order derivatives depends on the discretization used and often leads to non-optimal spread-out schemes.

Classical PDE Methods

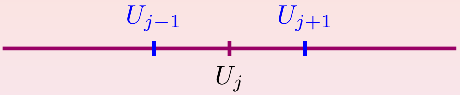
- Semi-discretization (or Method of Lines) is used to approximate PDEs by
  - firstly, discretize the spatial variables of PDEs to get a set of ODEs,
  - and then integrate along the time variable.
- However, many popular classical PDE methods are not MOL. Why?
- Two main disadvantages of MOL:
  - Stability is restricted by spatial discretization, possibly leading to unstable methods.
  - Approximation to higher order derivatives depends on the discretization used and often leads to non-optimal spread-out schemes.

Classical PDE Methods

- Semi-discretization (or Method of Lines) is used to approximate PDEs by
  - firstly, discretize the spatial variables of PDEs to get a set of ODEs,
  - and then integrate along the time variable.
- However, many popular classical PDE methods are not MOL. Why?
- Two main disadvantages of MOL:
  - Stability is restricted by spatial discretization, possibly leading to unstable methods.
  - Approximation to higher order derivatives depends on the discretization used and often leads to non-optimal spread-out schemes.

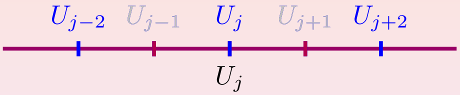
Classical PDE Methods

- Semi-discretization (or Method of Lines) is used to approximate PDEs by
  - firstly, discretize the spatial variables of PDEs to get a set of ODEs,
  - and then integrate along the time variable.
- However, many popular classical PDE methods are not MOL. Why?
- Two main disadvantages of MOL:
  - Stability is restricted by spatial discretization, possibly leading to unstable methods.
  - Approximation to higher order derivatives depends on the discretization used and often leads to non-optimal spread-out schemes.



Classical PDE Methods

- Semi-discretization (or Method of Lines) is used to approximate PDEs by
  - firstly, discretize the spatial variables of PDEs to get a set of ODEs,
  - and then integrate along the time variable.
- However, many popular classical PDE methods are not MOL. Why?
- Two main disadvantages of MOL:
  - Stability is restricted by spatial discretization, possibly leading to unstable methods.
  - Approximation to higher order derivatives depends on the discretization used and often leads to non-optimal spread-out schemes.



Classical PDE Methods

- Consider the advection/wave equation,

  $$\frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x} = 0 \quad \text{on the interval } (0,1) \text{ with } u(0,t) = u(1,t).$$

- By using central differences, we semi-discretize the PDE to an ODE system $d\mathbf{u}(t)/dt = A_h\mathbf{u}(t)$ with spatial stepsize $h = 1/N$, and then integrate the system by an explicit RK method with temporal stepsize δ. It follows that z^* must stay inside the stability region of the RK method to ensure the time integration is stable, where $z^* = \delta\lambda_k$, for $k = 1, \ldots, N$ and λ_k are the eigenvalues of A_h.

Classical PDE Methods

- Consider the advection/wave equation,

  $$\frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x} = 0 \quad \text{on the interval } (0,1) \text{ with } u(0,t) = u(1,t).$$

- By using central differences, we semi-discretize the PDE to an ODE system $d\mathbf{u}(t)/dt = A_h\mathbf{u}(t)$ with spatial stepsize $h = 1/N$, and then integrate the system by an explicit RK method with temporal stepsize δ. It follows that z^* must stay inside the stability region of the RK method to ensure the time integration is stable, where $z^* = \delta\lambda_k$, for $k = 1, \ldots, N$ and λ_k are the eigenvalues of A_h.

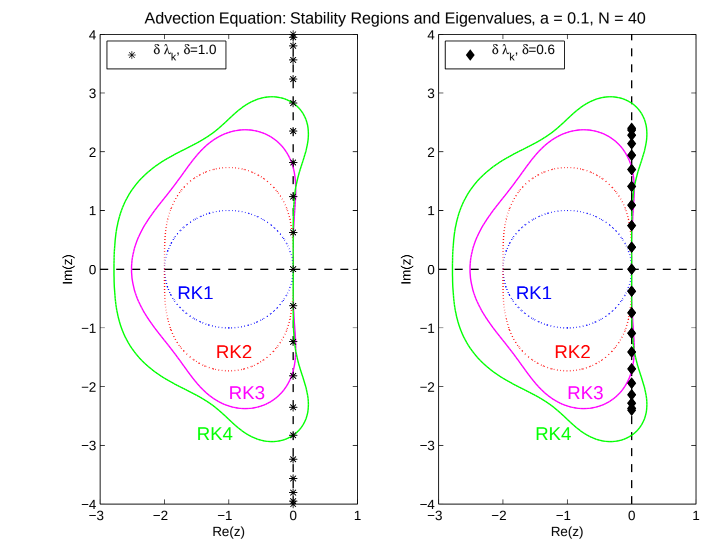
Classical PDE Methods

- Consider the advection/wave equation,

  $$\frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x} = 0 \quad \text{on the interval } (0,1) \text{ with } u(0,t) = u(1,t).$$

- By using central differences, we semi-discretize the PDE to an ODE system $d\mathbf{u}(t)/dt = A_h\mathbf{u}(t)$ with spatial stepsize $h = 1/N$, and then integrate the system by an explicit RK method with temporal stepsize δ. It follows that z^* must stay inside the stability region of the RK method to ensure the time integration is stable, where $z^* = \delta\lambda_k$, for $k = 1, \ldots, N$ and λ_k are the eigenvalues of A_h.

Classical PDE Methods – Method of Lines



A Novel Semi-Discretization Method

- We want to develop new discretization methods which overcome the disadvantages of MOL and unify MOL and other classical PDE methods under the same RK/TDRK structure.
- The idea is simple: we discretize the temporal variable t first. This means that the spatial discretization can then be chosen in a more flexible way to meet stability and/or computational requirements.
- Let $f(\eta)$ be a smooth function of η and we examine

$$\frac{\partial u}{\partial t} = f(\mathcal{P}(u)), \tag{1}$$

where $\mathcal{P}(u)$ be a linear partial differential operator with constant coefficients. For examples: $\mathcal{P}(u) = \frac{\partial}{\partial x} u$ and $\mathcal{P}(u) = \frac{\partial^2}{\partial x^2} u$.

A Novel Semi-Discretization Method

- We want to develop new discretization methods which overcome the disadvantages of MOL and unify MOL and other classical PDE methods under the same RK/TDRK structure.
- The idea is simple: we discretize the temporal variable t first. This means that the spatial discretization can then be chosen in a more flexible way to meet stability and/or computational requirements.
- Let $f(\eta)$ be a smooth function of η and we examine

$$\frac{\partial u}{\partial t} = f(\mathcal{P}(u)), \tag{1}$$

where $\mathcal{P}(u)$ be a linear partial differential operator with constant coefficients. For examples: $\mathcal{P}(u) = \frac{\partial}{\partial x} u$ and $\mathcal{P}(u) = \frac{\partial^2}{\partial x^2} u$.

A Novel Semi-Discretization Method

- We want to develop new discretization methods which overcome the disadvantages of MOL and unify MOL and other classical PDE methods under the same RK/TDRK structure.
- The idea is simple: we discretize the temporal variable t first. This means that the spatial discretization can then be chosen in a more flexible way to meet stability and/or computational requirements.
- Let $f(\eta)$ be a smooth function of η and we examine

$$\frac{\partial u}{\partial t} = f(\mathcal{P}(u)), \tag{1}$$

where $\mathcal{P}(u)$ be a linear partial differential operator with constant coefficients. For examples: $\mathcal{P}(u) = \frac{\partial}{\partial x} u$ and $\mathcal{P}(u) = \frac{\partial^2}{\partial x^2} u$.

A Novel Semi-Discretization Method

- Differentiate (1) with respect to t, we get

$$\frac{\partial^2 u}{\partial t^2} = f_\eta(\mathcal{P}(u))\mathcal{P}(f(\mathcal{P}(u))) = f_\eta \mathcal{P}(f).$$

- Compare with $\frac{d^2 y}{dt^2} = f_y f$ for $y'(t) = f(y)$.
- Similarly, we can derive all the higher derivatives and apply the tree theory for ODEs on PDEs.
- This enables us to apply ODE methods directly to PDEs.
- If we apply the explicit trapezoidal rule to the wave equation with appropriate compact schemes to approximate the spatial derivatives, we have a method which has order-2 behaviour in both time and space. In fact, this is the well-known Lax-Wendroff scheme.

A Novel Semi-Discretization Method

- Differentiate (1) with respect to t, we get

$$\frac{\partial^2 u}{\partial t^2} = f_\eta(\mathcal{P}(u))\mathcal{P}(f(\mathcal{P}(u))) \\ = f_\eta \mathcal{P}(f).$$

- Compare with $\frac{d^2 y}{dt^2} = f_y f$ for $y'(t) = f(y)$.
- Similarly, we can derive all the higher derivatives and apply the tree theory for ODEs on PDEs.
- This enables us to apply ODE methods directly to PDEs.
- If we apply the explicit trapezoidal rule to the wave equation with appropriate compact schemes to approximate the spatial derivatives, we have a method which has order-2 behaviour in both time and space. In fact, this is the well-known Lax-Wendroff scheme.

A Novel Semi-Discretization Method

- Differentiate (1) with respect to t, we get
$$\begin{aligned}\frac{\partial^2 u}{\partial t^2} &= f_\eta(\mathcal{P}(u))\mathcal{P}(f(\mathcal{P}(u))) \\ &= f_\eta \mathcal{P}(f).\end{aligned}$$
- Compare with $\dfrac{d^2 y}{dt^2} = f_y f$ for $y'(t) = f(y)$.
- Similarly, we can derive all the higher derivatives and apply the tree theory for ODEs on PDEs.
- This enables us to apply ODE methods directly to PDEs.
- If we apply the explicit trapezoidal rule to the wave equation with appropriate compact schemes to approximate the spatial derivatives, we have a method which has order-2 behaviour in both time and space. In fact, this is the well-known Lax-Wendroff scheme.

A Novel Semi-Discretization Method

- Differentiate (1) with respect to t, we get

$$\begin{aligned}\frac{\partial^2 u}{\partial t^2} &= f_\eta(\mathcal{P}(u))\mathcal{P}(f(\mathcal{P}(u))) \\ &= f_\eta \mathcal{P}(f).\end{aligned}$$

- Compare with $\dfrac{d^2 y}{dt^2} = f_y f$ for $y'(t) = f(y)$.
- Similarly, we can derive all the higher derivatives and apply the tree theory for ODEs on PDEs.
- This enables us to apply ODE methods directly to PDEs.
- If we apply the explicit trapezoidal rule to the wave equation with appropriate compact schemes to approximate the spatial derivatives, we have a method which has order-2 behaviour in both time and space. In fact, this is the well-known Lax-Wendroff scheme.

A Novel Semi-Discretization Method

- Differentiate (1) with respect to t, we get

$$\begin{aligned}\frac{\partial^2 u}{\partial t^2} &= f_\eta(\mathcal{P}(u))\mathcal{P}(f(\mathcal{P}(u))) \\ &= f_\eta \mathcal{P}(f).\end{aligned}$$

- Compare with $\dfrac{d^2 y}{dt^2} = f_y f$ for $y'(t) = f(y)$.
- Similarly, we can derive all the higher derivatives and apply the tree theory for ODEs on PDEs.
- This enables us to apply ODE methods directly to PDEs.
- If we apply the explicit trapezoidal rule to the wave equation with appropriate compact schemes to approximate the spatial derivatives, we have a method which has order-2 behaviour in both time and space. In fact, this is the well-known Lax-Wendroff scheme.

A Novel Semi-Discretization Method

- Differentiate (1) with respect to t, we get

$$\frac{\partial^2 u}{\partial t^2} = f_\eta(\mathcal{P}(u))\mathcal{P}(f(\mathcal{P}(u))) = f_\eta \mathcal{P}(f).$$

- Compare with $\dfrac{d^2y}{dt^2} = f_y f$ for $y'(t) = f(y)$.
- Similarly, we can derive all the higher derivatives and apply the tree theory for ODEs on PDEs.
- This enables us to apply ODE methods directly to PDEs.
- If we apply the explicit trapezoidal rule to the wave equation with appropriate compact schemes to approximate the spatial derivatives, we have a method which has order-2 behaviour in both time and space. In fact, this is the well-known Lax-Wendroff scheme.

An Application to PDE

- One popular method for solving PDE problems, such as the heat equation, is Crank-Nicolson method, which is an order-2 method.
- Heat Equation:

$$U_t = U_{xx},$$

  with I.C. $U(x, 0) = \sin(\pi x)$ and B.C. $U(0, t) = U(1, t) = 0$.
- Crank-Nicolson method: use 3-point second order approximation to U_{xx} and implicit midpoint or trapezoidal rule to solve the resulting tridiagonal system of ODEs.

An Application to PDE

- One popular method for solving PDE problems, such as the heat equation, is Crank-Nicolson method, which is an order-2 method.
- Heat Equation:

$$U_t = U_{xx},$$

with I.C. $U(x, 0) = \sin(\pi x)$ and B.C. $U(0, t) = U(1, t) = 0$.

- Crank-Nicolson method: use 3-point second order approximation to U_{xx} and implicit midpoint or trapezoidal rule to solve the resulting tridiagonal system of ODEs.

An Application to PDE

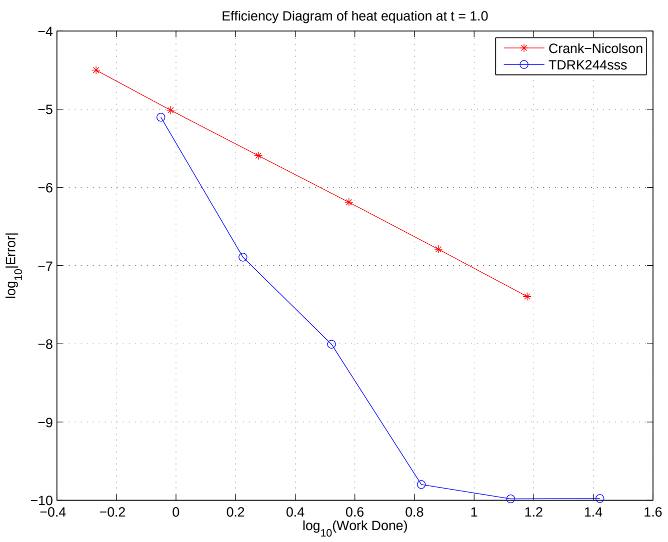
- One popular method for solving PDE problems, such as the heat equation, is Crank-Nicolson method, which is an order-2 method.
- Heat Equation:

$$U_t = U_{xx},$$

with I.C. $U(x,0) = \sin(\pi x)$ and B.C. $U(0,t) = U(1,t) = 0$.

- Crank-Nicolson method: use 3-point second order approximation to U_{xx} and implicit midpoint or trapezoidal rule to solve the resulting tridiagonal system of ODEs.

Heat Equation



Heat Equation and Order Reduction

- Practically, Crank-Nicolson method performs better than other higher order methods which suffer order reduction.
- Analyze the diffusion equation with non-homogeneous boundary values

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} \quad \text{on the interval } (0,1),$$

$$u(0,t) = p(t), \quad u(1,t) = 0, \quad \text{and} \quad u(x,0) = 0.$$

- After obtaining the semi-discrete system, we can decouple it to a equivalent non-homogenous ODE system which can then be written as the Prothero-Robinson equation.
- We conduct experiments for $p(t) = t^\alpha$ with different α values to compare the order behaviour of three methods: Crank-Nicolson, TDRK244sss and Gauss 2-stage methods.

Heat Equation and Order Reduction

- Practically, Crank-Nicolson method performs better than other higher order methods which suffer order reduction.
- Analyze the diffusion equation with non-homogeneous boundary values

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} \quad \text{on the interval } (0,1),$$
$$u(0,t) = p(t), \quad u(1,t) = 0, \quad \text{and} \quad u(x,0) = 0.$$

- After obtaining the semi-discrete system, we can decouple it to a equivalent non-homogenous ODE system which can then be written as the Prothero-Robinson equation.
- We conduct experiments for $p(t) = t^{\alpha}$ with different α values to compare the order behaviour of three methods: Crank-Nicolson, TDRK244sss and Gauss 2-stage methods.

Heat Equation and Order Reduction

- Practically, Crank-Nicolson method performs better than other higher order methods which suffer order reduction.
- Analyze the diffusion equation with non-homogeneous boundary values

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} \quad \text{on the interval } (0,1),$$
$$u(0,t) = p(t), \quad u(1,t) = 0, \quad \text{and} \quad u(x,0) = 0.$$

- After obtaining the semi-discrete system, we can decouple it to a equivalent non-homogenous ODE system which can then be written as the Prothero-Robinson equation.
- We conduct experiments for $p(t) = t^{\alpha}$ with different α values to compare the order behaviour of three methods: Crank-Nicolson, TDRK244sss and Gauss 2-stage methods.

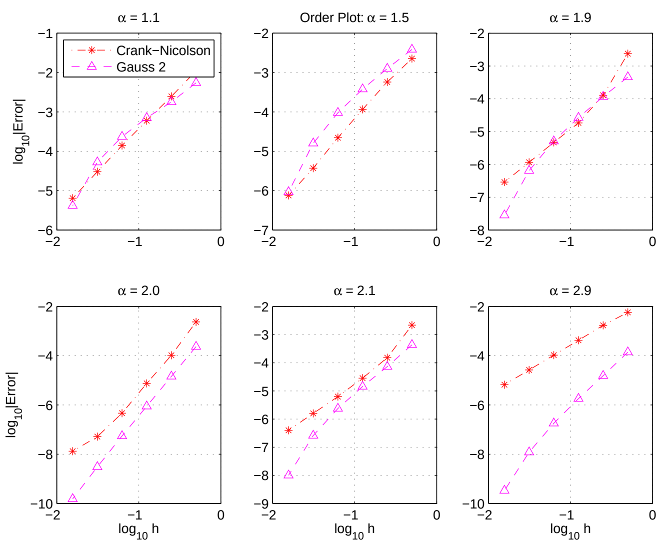
Heat Equation and Order Reduction

- Practically, Crank-Nicolson method performs better than other higher order methods which suffer order reduction.
- Analyze the diffusion equation with non-homogeneous boundary values

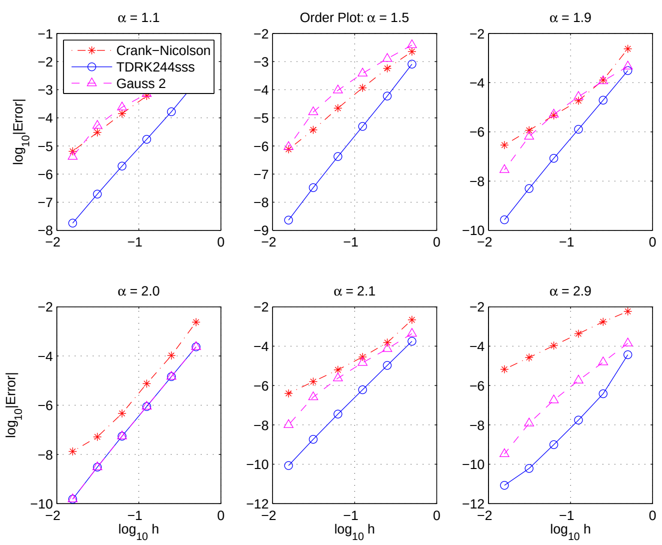
$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} \quad \text{on the interval } (0,1),$$
$$u(0,t) = p(t), \quad u(1,t) = 0, \quad \text{and} \quad u(x,0) = 0.$$

- After obtaining the semi-discrete system, we can decouple it to a equivalent non-homogenous ODE system which can then be written as the Prothero-Robinson equation.
- We conduct experiments for $p(t) = t^{\alpha}$ with different α values to compare the order behaviour of three methods: Crank-Nicolson, TDRK244sss and Gauss 2-stage methods.

Heat Equation with $u(0,t) = t^\alpha$



Heat Equation with $u(0,t) = t^\alpha$



Discussion/Conclusion

- TDRK methods are more efficient compared with some popular RK methods for the stiff problems we tested.
- The second derivative terms in TDRK give us more freedom and enable us to construct methods with higher stage order.
- Although the cost of calculating the second derivatives may be higher than the first derivatives, the advantage gained makes their use beneficial.
- For ODE problems: Our study suggests it will be of interest to implement a variable stepsize code for implicit TDRK methods.

Discussion/Conclusion

- TDRK methods are more efficient compared with some popular RK methods for the stiff problems we tested.
- The second derivative terms in TDRK give us more freedom and enable us to construct methods with higher stage order.
- Although the cost of calculating the second derivatives may be higher than the first derivatives, the advantage gained makes their use beneficial.
- For ODE problems: Our study suggests it will be of interest to implement a variable stepsize code for implicit TDRK methods.

Discussion/Conclusion

- TDRK methods are more efficient compared with some popular RK methods for the stiff problems we tested.
- The second derivative terms in TDRK give us more freedom and enable us to construct methods with higher stage order.
- Although the cost of calculating the second derivatives may be higher than the first derivatives, the advantage gained makes their use beneficial.
- For ODE problems: Our study suggests it will be of interest to implement a variable stepsize code for implicit TDRK methods.

Discussion/Conclusion

- TDRK methods are more efficient compared with some popular RK methods for the stiff problems we tested.
- The second derivative terms in TDRK give us more freedom and enable us to construct methods with higher stage order.
- Although the cost of calculating the second derivatives may be higher than the first derivatives, the advantage gained makes their use beneficial.
- For ODE problems: Our study suggests it will be of interest to implement a variable stepsize code for implicit TDRK methods.

Discussion/Conclusion

- We have developed a novel approach for the discretization of PDEs.
- This approach allows for more compact finite schemes for the higher derivatives and will provide a systematic way to apply ODE methods to PDEs.
- Many classical PDE schemes can be interpreted in the same way in terms of our new approach.
- The order-4 TDRK244sss method only requires twice the cost of the order-2 Crank-Nicolson method and is shown to be more efficient.
- We will further explore this type of numerical scheme for solving diffusion and diffusion-advection equations of higher dimension in the future.

Discussion/Conclusion

- We have developed a novel approach for the discretization of PDEs.
- This approach allows for more compact finite schemes for the higher derivatives and will provide a systematic way to apply ODE methods to PDEs.
- Many classical PDE schemes can be interpreted in the same way in terms of our new approach.
- The order-4 TDRK244sss method only requires twice the cost of the order-2 Crank-Nicolson method and is shown to be more efficient.
- We will further explore this type of numerical scheme for solving diffusion and diffusion-advection equations of higher dimension in the future.

Discussion/Conclusion

- We have developed a novel approach for the discretization of PDEs.
- This approach allows for more compact finite schemes for the higher derivatives and will provide a systematic way to apply ODE methods to PDEs.
- Many classical PDE schemes can be interpreted in the same way in terms of our new approach.
- The order-4 TDRK244sss method only requires twice the cost of the order-2 Crank-Nicolson method and is shown to be more efficient.
- We will further explore this type of numerical scheme for solving diffusion and diffusion-advection equations of higher dimension in the future.

Discussion/Conclusion

- We have developed a novel approach for the discretization of PDEs.
- This approach allows for more compact finite schemes for the higher derivatives and will provide a systematic way to apply ODE methods to PDEs.
- Many classical PDE schemes can be interpreted in the same way in terms of our new approach.
- The order-4 TDRK244sss method only requires twice the cost of the order-2 Crank-Nicolson method and is shown to be more efficient.
- We will further explore this type of numerical scheme for solving diffusion and diffusion-advection equations of higher dimension in the future.

Discussion/Conclusion

- We have developed a novel approach for the discretization of PDEs.
- This approach allows for more compact finite schemes for the higher derivatives and will provide a systematic way to apply ODE methods to PDEs.
- Many classical PDE schemes can be interpreted in the same way in terms of our new approach.
- The order-4 TDRK244sss method only requires twice the cost of the order-2 Crank-Nicolson method and is shown to be more efficient.
- We will further explore this type of numerical scheme for solving diffusion and diffusion-advection equations of higher dimension in the future.


Thank You