THREE- AND FOUR-DERIVATIVE HERMITE–BIRKHOFF–OBRECHKOFF SOLVERS FOR STIFF ODE

By
Njwd Albishi
February 2016

A Thesis
submitted to the Faculty of Graduate and Postdoctoral Studies
in partial fulfillment of the requirements
for the degree of
Master in Science in Mathematics

Abstract

Three- and four-derivative k-step Hermite–Birkhoff–Obrechkoff (HBO) methods are constructed for solving stiff systems of first-order differential equations of the form $y' = f(t, y)$, $y(t_0) = y_0$. These methods use higher derivatives of the solution y as in Obrechkoff methods. We compute their regions of absolute stability and show the three- and four-derivative HBO are $A(\alpha)$-stable with $\alpha > 71^\circ$ and $\alpha > 78^\circ$ respectively. We conduct numerical tests and show that our new methods are more efficient than several existing well-known methods.

Key words: general linear method for stiff ODE's; Hermite-Birkhoff-Obrechkoff method; maximum end error; number of function evaluations; CPU time; comparing stiff ODE solvers.

Acknowledgements

During the course of my Masters degree at the University of Ottawa I have had to meet and overcome many challenges and obstacles. Without the help of several people, and first of all God (Allah) first, It would have been much harder and more stressful here. I was able to be successful in my program primarily with the support of my supervisors, whom I would like to thank from the bottom of my heart. Both Drs. **Rémi Vaillancourt** and **Thierry Giordano** were instrumental in my academic success at the University of Ottawa. I would also like to give special thanks to **Dr. Troung Nguyen-Ba**, who helped me throughout the entire duration of my program, following, motivating, and advising me step by step during my studies. Of course, I would also like to thank my mom who was an excellent role model for me while I was growing up and who encouraged me to become the person that I am today. I want to thank my husband, who supported me, both in my original decision to come to Canada and in my pursuit of my Masters degree at the University of Ottawa. Without coming home to him each day I wouldn't have had the stability that I needed to be able to deal with the stresses of completing a Masters degree. I would also like to thank my brother and my six sisters who kept praying for me during my time here in Canada. They will always give me wisdom and positive influence during times of need. My friends, both those that I left behind in Saudi Arabia, and those that I made here in Canada, were all very supportive throughout my education here in Canada. Finally I would like to thank the government of the Kingdom of Saudi Arabia for their financial support and for providing me with the opportunity to study abroad. I know that not everyone has this kind of an opportunity and I am truly grateful to be able to take advantage of this study abroad program. The program allows Saudi

youth to experience different cultures, learn a different language, and take valuable knowledge back to Saudi Arabia for the education of Saudis who did not have the same opportunity.

Dedication

I would like to dedicate my thesis to my family. They are the one constant in my life who have always been at my side throughout my academic development. My mom, **Sarraa**, my husband, **Mohammed**, my brother, **Abd Al-Aziz** and my sisters **Ohoud**, **Abeer**, **Ghadeer**, **Shorooq**, **Shumookh**, and **Hatun**; without them I don't know where I would be in life, with them I know that I will be able to succeed at anything. I also want to dedicate this to my children, whom I hope will someday read this and know that they meant so much to me even while I was concentrating on my Masters degree in a far away place; **Meshari** my son and **Shaden** my daughter.

Contents

Abstract **ii**

Acknowledgements **iii**

Dedication **v**

1 Introduction **1**

1.1 Organization of the Thesis 2

1.2 Thesis contribution 2

2 Background material review **4**

2.1 Linear multistep methods 4

2.1.1 Numerical methods: notation 4

2.1.2 Linear multistep methods: notations 5

2.2 Linear stability 6

2.3 Stiff differential equations 9

2.3.1 Some characterization of stiffness 10

2.4 Example of linear multistep methods for stiff ODEs 11

2.4.1 Backward differentiation formula 11

2.4.2 Numerical differentiation formula 12

2.4.3 Enright's two-derivative method 14

2.4.4 Ezzeddine *et al.*'s third derivative multistep methods 16

2.4.5 Other known methods 16

3 Three-derivative methods **17**
3.1 HBO(3,p); $5 \leq p \leq 14$. 17
3.2 Regions of absolute stability 18
3.3 Principal error term 21
3.4 Numerical Results 22
3.4.1 Iteration scheme 22
3.4.2 Implementation and problems used for comparison 23
3.4.3 Comparing CPU time on the van der Pol oscillator 24
3.4.4 Comparing CPU time on the Robertson chemical reaction problem 28
3.4.5 Comparing CPU time on a stiff DETEST problem 30

4 Four-derivative methods **35**
4.1 HBO(4,p); $7 \leq p \leq 14$. 35
4.2 Regions of absolute stability 36
4.3 The principal error term 38
4.4 Numerical Results 39
4.4.1 Iteration scheme 39
4.4.2 Implementation and problems used for comparison 40
4.4.3 Comparing CPU time on the van der Pol oscillator 41
4.4.4 Comparing CPU time on the Oregonator describing Belusov–Zhabotinskii reaction 42
4.4.5 Comparing CPU time on a stiff DETEST problem 45

5 Conclusion **50**

6 Coefficients for HBO$(3,p)$ and HBO$(4,p)$ **52**
6.1 Coefficients of new 3-derivative HBO$(3,p)$ and 4-derivative HBO$(4,p)$ 52
6.1.1 Coefficients of HBO$(3,p)$, of order $p = 5, 6, \ldots, 13$. 52
6.1.2 Coefficients of HBO$(4,p)$, of order $p = 5, 6, \ldots, 13$. 53

List of Figures

1 Upper parts of the regions of absolute stability for k-step BDF for $k = 1, 2 \ldots, 6$. These regions include the negative real axis. 13

2 Regions of absolute stability, $\mathcal{R}$, of HBO(3,5–14). 20

3 HBO$(3, p)$, $p = 9, 11, 13$, are compared with SDMM(9) (left) and BDF(5) (right) for Problem (vdP) 25

4 straight line that best fits the data $(\log_{10}(\text{EPE}), \log_{10}(\text{CPU}))$ of HBO(3,9) solving Problem (vdP) 26

5 HBO$(3, p)$, $p = 9, 11, 13$, are compared with TDMM(9) (top left), TDMM(11) (top right) and TDMM(13) (bottom) for Problem (vdP). 27

6 HBO(3,9), HBO(3,11) and HBO(3,13) are compared with SDMM(9) (left), and BDF(5) (right) for Problem (RC). 29

7 HBO(3,9), HBO(3,11) and HBO(3,13) are compared with TDMM(9) (top left), TDMM(11) (top right) and TDMM(13) (bottom) for Problem (RC). 31

8 HBO$(3, p)$, $p = 9, 11, 13$, are compared with SDMM(9) (left) and BDF(5) (right) for Problem (D1). 32

9 HBO$(3, p)$, $p = 9, 11, 13$, are compared with TDMM(9) (top left), TDMM(11) (top right) and TDMM(13) (bottom) for Problem (D1). . 34

10 Regions of absolute stability, $\mathcal{R}$, of HBO(4,7–14). 37

11 HBO$(4, p)$, $p = 9, 11, 13$, are compared with SDMM(9) (left) and BDF(5) (right) for Problem (vdP). 41

12 HBO$(4,p)$, $p = 9, 11, 13$, are compared with TDMM(9) (top left), TDMM(11) (top right) and TDMM(13) (bottom) for Problem (vdP). 43
13 HBO$(4,p)$, $p = 9, 11, 13$, are compared with SDMM(9) (left) and BDF(5) (right) for Problem (OdB) 44
14 HBO$(4,p)$, $p = 9, 11, 13$, are compared with TDMM(9) (top left), TDMM(11) (top right) and TDMM(13) (bottom) for Problem (OdB). 46
15 HBO$(4,p)$, $p = 9, 11, 13$, are compared with SDMM(9) (left) and BDF(5) (right) for Problem (D1). 47
16 HBO$(4,p)$, $p = 9, 11, 13$, are compared with TDMM(9) (top left), TDMM(11) (top right) and TDMM(13) (bottom) for Problem (D1). . 48

List of Tables

1 Coefficients of the BDF methods of order 1 to 6. 12
2 Coefficients of the NDF methods of order 1 to 5. 14
3 Coefficients of the TDMM methods of order 4 to 8. 16

4 For a given order p, the table lists the angles α of $A(\alpha)$-stability for HBO$(3,p)$, SDMM(p) and TDMM(p), respectively. 19
5 For a given order p, the table lists the PLTC of HBO$(3,p)$ and SDMM(p), respectively. 21
6 CPU PEG and NST PEG of the listed HBO$(3,p)$ over SDMM(9) and BDF(5) for Problem (vdP). 27
7 CPU PEG and NST PEG of the listed HBO$(3,p)$ over TDMM(p) for Problem (vdP). 28
8 CPU PEG and NST PEG of HBO$(3,p)$, $p = 9, 11, 13$ methods over SDMM(9) and BDF(5) for Problem (RC). 29
9 CPU PEG and NST PEG of the listed HBO$(3,p)$ over TDMM(p) for Problem (RC). 30
10 CPU PEG and NST PEG of HBO$(3,p)$, $p = 9, 11, 13$, over BDF(5) and SDMM(9) for Problem (D1). 32
11 CPU PEG and NST PEG of the listed HBO$(3,p)$ over TDMM(p) for Problem (D1). 33

12 For a given order p, the table lists the angles α of $A(\alpha)$-stability for HBO$(4,p)$ and SDMM(p), respectively. 38

13 For a given order p, the table lists the PLTC of HBO$(4,p)$ and SDMM(p), respectively. 39
14 CPU PEG and NST PEG of the listed HBO$(4,p)$ over SDMM(9) and BDF(5) for Problem (vdP). 42
15 CPU PEG and NST PEG of the listed HBO$(4,p)$ over TDMM(p), for Problem (vdP). 42
16 CPU PEG and NST PEG of listed HBO$(4,p)$, $p = 9, 11, 13$, methods over SDMM(9) and BDF(5) for Problem (OdB). 45
17 CPU PEG and NST PEG of the listed HBO$(4,p)$ over TDMM(p) for Problem (OdB). 45
18 CPU PEG and NST PEG of HBO$(4,p)$, $p = 9, 11, 13$, over BDF(5) and SDMM(9) for Problem (D1). 47
19 CPU PEG and NST PEG of the listed HBO$(4,p)$ over TDMM(p) for Problem (D1). 49

20 Coefficients of HBO$(3,p)$ of order $p = k + 4 = 5, 6, 7$. 52
21 Coefficients of HBO$(3,p)$ of order $p = 8, 9, 10$. 53
22 Coefficients of HBO$(3,p)$ of order $p = 11, 12, 13$. 54
23 Coefficients of HBO$(4,p)$ of order $p = k + 6 = 7, 8, 9$. 54
24 Coefficients of HBO$(4,p)$ of order $p = 10, 11, 12$. 55
25 Coefficients of HBO$(4,p)$ of order $p = 13$. 55

Chapter 1

Introduction

In this thesis, we generalize explicit Obrechkoff methods defined and studied in [34] to implicit k-step, three- and four-derivative Hermite–Birkhoff–Obrechkoff methods. We will denote by HBO$(3,p)$ (respectively HBO$(4,p)$) the three (respectively four) derivative methods, of order $5 \leq p \leq 14$ (respectively $7 \leq p \leq 14$). These new methods are named Hermite–Birkhoff–Obrechkoff as they use Hermite–Birkhoff interpolation polynomials and values from y' to $y^{(4)}$ like Obrechkoff methods. HBO(d,p), are designed for solving stiff systems of first-order initial value problems

$$y' = f(t,y), \qquad y(t_0) = y_0, \qquad \text{where} \quad ' = \frac{d}{dt}, \tag{1}$$

where f is smooth enough to be able to compute y'', y''', $\ldots$, $y^{(d)}$. The derivative of y are computed either analytically or recursively. The recursive computation of higher derivatives using Taylor coefficients was used for example by Steffensen [41], Rabe [38] and others (see, for instance, [23, pp. 46–49] and [5]). Deprit and Zahar [12] showed that recursive computation of Taylor coefficients is very effective in achieving high accuracy with little computing time and large step sizes.

In applications, one meets many stiff problems such as (1), for instance, nonlinear chemical problems (Robertson, 1966), chemical pyrolysis (Datta, 1967).

The increased efficiency of our methods HBO$(3,p)$ and HBO$(4,p)$, for $p \leq 14$ is achieved by the addition of high order derivatives even with the evaluation of Taylor coefficients of the functions involved. We compare our methods with the well-known

and frequently used methods:

- BDF(p): Gear backward differentiation methods of order $p \leq 6$, (see Subsection 2.4.1).
- SDMM(p): Enright second derivative multistep methods of order $p \leq 9$, (see Subsection 2.4.3).
- TDMM(p): Ezzeddine *et al.* third derivative multistep methods of order $p \leq 14$, (see Subsection 2.4.4).

1.1 Organization of the Thesis

This thesis consists of six chapters. The introduction and the list of contributions are the first one. In Chapter 2, we present a brief summary about known methods for solving a stiff ODE system. Chapter 3 is divided in two parts. In the former part, we present three-derivative Hermite-Birkhoff-Obrechkoff methods, their regions of absolute stability and principal error terms. In the later part, we present our numerical results, which include an iteration scheme, and compare CPU time on the following test problems: the van der Pol oscillator (Subsection (3.4.3)), the Robertson chemical reaction problem (Subsection(3.4.4)) and the stiff DETEST problem D1 (Subsection (3.4.5)). In Chapter 4, we first, present four-derivative Hermite-Birkhoff-Obrechkoff methods, their regions of absolute stability and principal error terms; then, we present numerical results, which include an iteration scheme, and compare CPU time on the following test problems: the van der Pol oscillator, the Oregonator describing the Belusov-Zhabotinskii reaction (Subsection (4.4.4)) and the stiff DETEST problem D1. The conclusion of the thesis is presented in Chapter 5. The coefficients of the new HBO$(3, p)$ and HBO$(4, p)$ are given in the Appendix (Chapter 6).

1.2 Thesis contribution

We hope that the new results we have obtained will become a very useful development of stiff ODE solvers. We summarize below our contributions:

- We introduce three- and four-derivative k-step Hermite–Birkhoff–Obrechkoff methods of order 5 to 14 and 7 to 14 respectively and compute their regions of absolute stability.
- We show that for $p = 9, 11$ and 13, HBO$(3, p)$ compare favorably with BDF(p) and SDMM(p). Our methods give the best result in C++.
- We show that for $p = 9, 11$ and 13, HBO$(3, p)$ compare favorably with TDMM(p). Our methods give a good result in MATLAB.
- We show that for $p = 9, 11$ and 13, HBO$(4, p)$ compare favorably with BDF(p) and SDMM(p). Our methods give a good result in C++.
- We show that for $p = 9, 11$ and 13, HBO$(4, p)$ compare favorably with TDMM(p). Our methods give a good result in MATLAB.

Chapter 2

Background material review

In this chapter, we summarize the background material used in this thesis. In Section 2.1, we present briefly linear multistep methods and in Section 2.2, we introduce linear stability. We discuss stiff ODEs in Section 2.3 and present in Section 2.4 some well known numerical methods for solving them. The material presented in this chapter can be found in [32], [24] and [43].
For Section 2.3, we have also used references [1], [2] and [48].

2.1 Linear multistep methods

2.1.1 Numerical methods: notation

Consider an initial value problem:

$$y' = f(t, y), \qquad y(t_0) = y_0,$$

as in the Introduction. Numerical methods are techniques to solve initial value problems on the interval $[t_0, t_{end}]$, $t_{end} < \infty$, by finding, for $n = 0, 1, 2, \cdots, N$, approximate values y_n of the exact solution $y(t_n)$, where $t_n = t_0 + nh$ and $h = (t_{end} - t_0)/N$. The parameter h is called the step size.
For example the two simplest numerical methods to solve initial value problem are

- The explicit Forward Euler method

$$y_{n+1} = y_n + f(t_n, y_n) \tag{2}$$

- The implicit Backward Euler method

$$y_{n+1} = y_n + f(t_{n+1}, y_{n+1}) \tag{3}$$

The first method is *explicit* as, given y_n, the difference equation (2) yields y_{n+1} explicitly. In the second method, y_{n+1} cannot be computed without solving the implicit equation (3). This method is *implicit*. The HBO methods we construct below are implicit.

2.1.2 Linear multistep methods: notations

A general k-step linear multistep method may be written

$$\sum_{j=0}^{k} \alpha_j y_{n+j} = h \sum_{j=0}^{k} \beta_j f\left(t_{n+j}, y_{n+j}\right), \tag{4}$$

where α_j and β_j are constant subject to the conditions

$$\alpha_k = 1 \qquad \text{and} \qquad |\alpha_0| + |\beta_0| \neq 0.$$

Applied to the test problem $y' = \lambda y$, Equ. (4) becomes

$$\sum_{j=0}^{k} \alpha_j y_{n+j} = h\lambda \sum_{j=0}^{k} \beta_j y_{n+j},$$

and gives the following difference equation

$$\sum_{j=0}^{k} (\alpha_j - h\lambda\beta_j) y_{n+j} = 0. \tag{5}$$

Equ. (5) is a complex linear, constant coefficient difference equation that can be solved by setting $y_{n+j} = \xi^{n+j}$ and solving the resulting characteristic polynomial (with $\hat{h} = h\lambda$)

$$\pi(r, \hat{h}) = \sum_{j=0}^{k} (\alpha_j - \hat{h}\beta_j)\, r^j = 0. \tag{6}$$

The solution of Equ. (5) is given by

$$y_n = \sum_{i=0}^{l} p_i(n)\,\xi_i^n$$

where $l \leq k$ and ξ_i, $i = 1, \ldots, k$ and the distinct roots of Equ. (6) and each p_i is a polynomial in n of degree one less the multiplicity of ξ_i.

Example

- The explicit Forward Euler method is a 1-step linear method with $\beta_1 = 0, \alpha_1 = \beta_0 = 1$ and $\alpha_0 = -1$.
- The implicit Backward Euler method is a 1-step linear method with $\beta_0 = 0, \alpha_1 = \beta_1 = 1$ and $\alpha_0 = -1$.

2.2 Linear stability

Let $y' = f(t, y), \quad y(t_0) = y_0$ be an initial value problem as in Equ. (1) and let us assume that its exact solution displays certain stability properties. Then a numerical method will be stable if it replicates the long-term dynamics properties of Equ. (1). The analysis of the long-term behaviour of numerical methods for initial value problems begins with a study of a scalar linear, constant coefficient, test problem. Throughout this section we consider the linear problem

$$y' = \lambda y, \qquad \lambda \in \mathbb{C}, \qquad \mathrm{Re}(\lambda) \leq 0. \tag{7}$$

Let us first recall the definition of absolute stability (see Remark 1).

Definition 1 *([43], Definition 3.6.1) The region of absolute stability $\mathcal{R}$ of a k-step numerical method for the solution of Equ.* (1) *is the set of points $\hat{h} = h\lambda$ in the complex plane with the property that if $\hat{h} \in \mathcal{R}$, then there is a constant C such that the numerical method applied to Equ.* (7) *satisfies*

$$\sup_{n \geq 0} \|y_n\| \leq C \max_{0 \leq j \leq k-1} \|y_j\|$$

for all initial data $\{\|y_j\|\,; 0 \leq j \leq k-1\}$.

Proposition 1 ([43], Lemma 3.6.3)
For a linear multistep method (4), the region of absolute stability is the subset $\mathcal{R}$ of all points $\hat{h} \in \mathbb{C}$ such that all the roots of the characteristic polynomial $\pi(r, \hat{h})$ lie inside or on the unit circle and those on the unit circle are simple.

Remark 1
In [32], Lambert uses a stronger notion of absolute stability. Indeed, he restricts in Equ. (7) to the case of linear decay (i.e. $\mathrm{Re}\,\lambda < 0$) and a linear multistep method will be absolutely stable if $\|y_n\| \to 0 \quad \text{as} \quad n \to \infty$. Hence, the region of absolute stability $\mathcal{R}$ will become the subset of all points $\hat{h} \in \mathbb{C}$ such that all the roots of $\pi(r, \hat{h})$ lie inside the unit circle.
From now on, we will use the (stronger) notion of absolute stability given by Lambert in [32] (see [43], Chapter 5, for a more general approach).
In many circumstances, we want the numerical solution to replicate the long term behaviour of the exact solution without restriction on h, hence the following definition.

Definition 2 *A linear multistep method is said to be A-stable if the region of absolute stability* $\mathcal{R}$ *satisfies*

$$\left\{\hat{h} \in \mathbb{C}; \mathrm{Re}(\hat{h}) < 0\right\} \subset \mathcal{R}.$$

Example

1. The simplest A-stable Runge-Kutta method is the backward Euler method (see for example [43, pp. 362–363]).

2. The backward differentiation formula of order 2 (BDF2) is A-stable (see for example [32, pp. 98–101]).

A-stability is a strong requirement, and is too strong for some problems. By restricting the class of test problems, we can consider a weaker requirement, which will remove the restriction on the step size for this class of problems. Consider for example the linear, constant coefficient system

$$y' = Ay, \quad y(0) = y_0, \tag{8}$$

where the spectrum of A lies inside a wedge entirely contained in the left half-plane and making angles α with the real axis. For such problems, the following definition of stability is natural.

Definition 3 *A linear-multistep method is $A(\alpha)$-stable for $\alpha \in \left(0, \frac{\pi}{2}\right)$ if*

$$\left\{\hat{h} \in \mathbb{C}; \pi - \alpha < \arg(\hat{h}) < \pi + \alpha\right\} \subseteq \mathcal{R}.$$

*It is $A\,(0)$-**stable** if it is $A(\alpha)$-stable for some $\alpha \in (0, \pi/2)$.*

Example The backward differentiation formula BDF(p) are $A(\alpha)$-stable for $p \leq 6$ ([32, pp. 98–101]).

Remark 2

1. - A $A(\alpha)$-stable method is $A(\beta)$-stable for all $0 \leq \beta \leq \alpha$,
2. - A method is A-stable if and only if it is $A(\alpha)$-stable, for all $0 < \alpha < \frac{\pi}{2}$,
3. - For all $\alpha \in \left(0, \frac{\pi}{2}\right)$, the region of absolute stability of a $A(\alpha)$-stable method contains $\left\{\hat{h} \in \mathbb{C};\ \mathrm{Re}(\hat{h}) < 0 \quad \text{and} \quad \mathrm{Im}(\hat{h}) = 0\right\}$.

This last remark motivates the following definition ([32, pp. 225]).

Definition 4 *A linear method is said to be A_0-**stable** if*

$$\left\{\hat{h} \in \mathbb{C};\ \mathrm{Re}(\hat{h}) < 0 \quad \textit{and} \quad \mathrm{Im}(\hat{h}) = 0\right\} \subseteq \mathcal{R}.$$

Other notions of stability have been considered. As we do not pursue their study in this thesis, we state them only for completeness.

Definition 5 *A linear multistep method is **stiffly stable** with real positive parameters a and c if $\mathcal{R}_1 \cup \mathcal{R}_2 \subseteq \mathcal{R}$ where*

$$\mathcal{R}_1 = \left\{\hat{h} \mid \mathrm{Re}(\hat{h}) < -a\right\}$$

and

$$\mathcal{R}_2 = \left\{\hat{h} \mid -a \leq \mathrm{Re}(\hat{h}) < 0, -c \leq \mathrm{Im}(\hat{h}) \leq c\right\}.$$

Remark 3 For a linear multistep method, we have the following hierarchy:

1. A-stable $\Rightarrow$ stiffly stable $\Rightarrow$ $A(\alpha)$-stable for some $\alpha \in (0, \pi/2) \Rightarrow$ $A(0)$-stable $\Rightarrow$ A_0-stable.

A stronger concept of stability for linear scalar test problem is the following. It uses the notion of the stability function R of a numerical method. We refer the reader to ([32, pp. 199]) for the precise definition of the stability function.

Definition 6 *A one-step method is said to be L-stable (respectively $L(\alpha)$-stable) if it is A-stable (respectively $A(\alpha)$-stable) and when applied to the scalar test problem* $y' = \lambda y$, $\mathrm{Re}(\lambda) < 0$,

$$|R(\lambda h)| \to 0 \quad as \quad h \to \infty.$$

A generalization of L-stability (respectively $L(\alpha)$-stability) to linear multistep method was given for example in [11], [25] and [30].
Note that the methods we develop in Chapter 3 and 4 are $L(\alpha)$-stable.

Example The Backward Euler Method is a L-stable, one step method, as its stability function is $R(z) = \frac{1}{1-z}$.

2.3 Stiff differential equations

A stiff equation is a differential equation for which certain numerical methods are numerically unstable, unless the step size is extremely small [48]. There is no precise definition of stiffness. Following [24], let us recall the first characterization, given by Curtiss and Hirschfelder, in 1952: "stiff equations are equations where certain implicit methods, in particular BDF, perform better, usually tremendously better, than explicit ones". The system introduced by Robertson in 1966, of a chemical reaction conferring fast and slow reaction is a well-known example of a stiff system of ODEs:

$$\begin{aligned} y_1' &= -0.04y_1 + 10^4 y_2 y_3, \\ y_2' &= 0.04y_1 - 10^4 y_2 y_3 - 3 \times 10^7 y_2^2, \\ y_3' &= 3 \times 10^7 y_2^2. \end{aligned} \tag{9}$$

If one treats this system on a short interval, e.g. $t \in [0, 40]$ there is no problem in numerical integration. However, if the interval is very large, then many standard codes fail to integrate it correctly [48]. We will study in details this system of ODEs in Section 3.4.4.

2.3.1 Some characterization of stiffness

As already indicated, there is no precise definition of stiffness, Hairer and Wanner mentioned, in [24, p. 1], that the most pragmatic characterization is also the first one, given by Curtiss and Hirschfelder.
Given initial value problem

$$y' = f(t, y), \quad y(t_0) = y_0,$$

consider the $(m \times m)$-Jacobian matrix

$$J = \left(\frac{\partial f_i}{\partial y_j} \right)_{1 \le i,j \le m}. \tag{10}$$

We assume that the m eigenvalues $\lambda_1, \ldots, \lambda_m$ of the matrix J are ordered as follows:

$$\operatorname{Re} \lambda_m \le \ldots \operatorname{Re} \lambda_2 \le \operatorname{Re} \lambda_1 < 0. \tag{11}$$

The following definition occurs in discussing stiffness.

Definition 7 *The stiffness ratio of the system $y' = f(t, y)$ is the positive number*

$$r = \frac{\operatorname{Re} \lambda_m}{\operatorname{Re} \lambda_1},$$

where the eigenvalues of the Jacobian matrix J of the system satisfy the relations (11).
The phenomenon of stiffness appears under various aspects (see [32], p 216–217).

- A linear constant coefficient system is stiff if all its eigenvalues have negative real parts and its stiffness ratio is large.
- Stiffness occurs when stability requirements, rather than those of accuracy, constrain the step size.

- Stiffness occurs when some components of the solution decay much more rapidly than others.
- A system is said to be stiff in a given interval I containing t if in I the neighbouring solution curves approach the solution curve at a rate which is very large in comparison with the rate at which the solution varies in that interval.

A statement that we take as a definition of stiffness due to Lambert is one which merely relates what is observed happening in practice.

Definition 8 *(see [32], p 220) If a numerical method with a finite region of absolute stability, applied to a system with any initial conditions, is forced to use, in a certain interval of integration, a step size which is excessively small in relation to the smoothness of the exact solution in that interval, then the system is said to be* ***stiff*** *in that interval.*

2.4 Example of linear multistep methods for stiff ODEs

In the following subsections, we present different numerical methods mentioned in the thesis. We start with the backward differentiation formulas, denoted by BDF, and their more recent versions: the numerical differentiation formulas, denoted by NDF. We will compare these linear multistep methods with the new methods we built in Chapter 3 and Chapter 4.

2.4.1 Backward differentiation formula

The first use of BDF methods appears to date back to Curtiss and Hirschfelder (1952), although then they were not given that name. Curtiss and Hirschfelder said "stiff equations are equations where certain implicit methods, and in particular backward-differentiation formulas (BDFs), perform better, usually tremendously better, than explicit ones". The importance of the BDF-based methods is their stability: they are stable along the entire negative real axis. This makes them suitable for solving

stiff equations. With their superior stability properties, they can be used with much larger step sizes than explicit methods.

In [21, 22], Gear constructed a series of stiffly-stable backward differentiation formula of order k up to six, denoted by BDF(k):

$$\sum_{j=0}^{k} \alpha_j y_{n+j} = h\beta_k f_{n+k}. \tag{12}$$

BDF methods of order larger than 6 are unstable. Indeed, Cryer in [9] proved that "The backward-difference multistep method BDF(k) satisfies the root condition iff $1 \leq k \leq 6$". The coefficients of the BDF methods are listed in Table 1.

BDF methods are also among the most efficient linear multistep methods for stiff differential equations ([45] and [36]).

The upper left parts of the regions of absolute stability of the BDF methods are the exterior of closed regions. Upper parts of the regions of absolute stability for k-step BDF for $k = 1, 2, 3$ (left panel), $k = 4, 5, 6$ (right panel) are shown in Fig. 1.

Table 1: Coefficients of the BDF methods of order 1 to 6.

k	α_6	α_5	α_4	α_3	α_2	α_1	α_0	β_k	p	C_{p+1}	α
1						1	-1	1	1	1	90°
2					1	$-\frac{4}{3}$	$\frac{1}{3}$	$\frac{2}{3}$	2	$-\frac{2}{9}$	90°
3				1	$-\frac{18}{11}$	$\frac{9}{11}$	$-\frac{2}{11}$	$\frac{6}{11}$	3	$-\frac{3}{22}$	86°
4			1	$-\frac{48}{25}$	$\frac{36}{25}$	$-\frac{16}{25}$	$\frac{3}{25}$	$\frac{12}{25}$	4	$-\frac{12}{125}$	73°
5		1	$-\frac{300}{137}$	$\frac{300}{137}$	$-\frac{200}{137}$	$\frac{75}{137}$	$-\frac{12}{137}$	$\frac{60}{137}$	5	$-\frac{110}{137}$	51°
6	1	$-\frac{360}{147}$	$\frac{450}{147}$	$-\frac{400}{147}$	$\frac{225}{147}$	$-\frac{72}{147}$	$\frac{10}{147}$	$\frac{60}{147}$	6	$-\frac{20}{343}$	18°

2.4.2 Numerical differentiation formula

Numerical differentiation formulas (NDF) are a modification of Backward differentiation formulas (BDF). We can rewrite the BDF formula of order k in Equ. (12) as

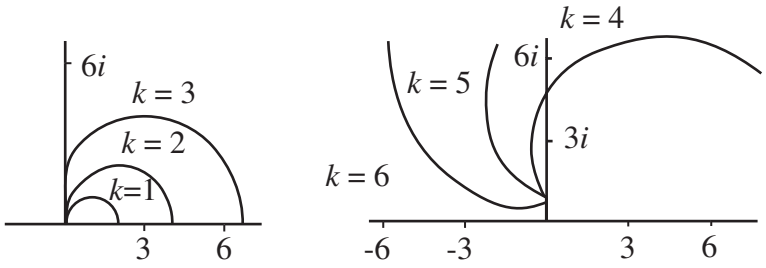


Figure 1: Upper parts of the regions of absolute stability for k-step BDF for $k = 1, 2 \ldots, 6$. These regions include the negative real axis.

follows,

$$\sum_{m=1}^{k} \frac{1}{m} \bigtriangledown^m y_{n+1} - hf(t_{n+1}, y_{n+1}) = 0, \tag{13}$$

where $\bigtriangledown^m$ denotes backward differences:

$$\begin{aligned} \bigtriangledown^m y_{n+1} &= \bigtriangledown^{m-1} (\bigtriangledown y_{n+1}) \\ &= \bigtriangledown^{m-1} (y_{n+1} - y_n) . \end{aligned}$$

A simplified Newton (chord) iteration solves the algebraic equation for y_{n+1}. This iteration is started with the predicted value.

$$y_{n+1}^{(0)} = \sum_{m=0}^{k} \bigtriangledown^m y_n. \tag{14}$$

Since the predictor Equ. (14) has a longer memory (needs more backsteps) than Equ. (13), Klopfenstein [28] and Reiher [40] decided to study NDF(k) formulas of the form

$$\sum_{m=1}^{k} \frac{1}{m} \bigtriangledown^m y_{n+1} - hf(t_{n+1}, y_{n+1}) - \kappa\gamma_k \left(y_{n+1} - y_{n+1}^0\right) = 0, \tag{15}$$

where κ is a scalar parameter and the coefficient γ_κ is equal to $\gamma_\kappa = \sum_{j=1}^{k} \frac{1}{j}$. Klopfenstein and Reiher [28] found numerically that the scalar κ widens the angle of $A(\alpha)$-stability for order 3 to 6. Because BDF(2) is already A-stable, Klopfenstein and Reiher tried to make an optimal choice for the scalar κ so that it could reduce the truncation error as much as possible and still retaining A-stability. Klopfenstein and

Reiher's optimal choice is $\kappa = -\frac{1}{9}$, giving a truncation error coefficient half that of BDF(2). This implies that, for sufficiently small step sizes, NDF(2) can have the same accuracy as BDF(2) with a step size about 26% bigger. Klopfenstein and Reiher's formulas are less successful at order higher than 2 because the price to pay to improve stability is a reduced efficiency. On the other hand, Shampine and Reichelt [44] looked at values of the scalar parameter κ that would make NDFs more accurate than BDFs and not much less stable. Since Klopfenstein's second order formula increases accuracy while retaining L-stability, this formula serves as the order 2 method of the new NDF family proposed by Shampine and Reichelt. Correspondingly, these authors looked for obtaining the same improvement in efficiency (26%) at orders 3 to 5. Because the stability of BDF(5) is so poor, Shampine and Reichelt were not willing to reduce its truncation error at all. The sufficient condition for NDF(1) to be A-stable is $1 - 2\kappa \geq 0$ and, for $1 < p \leq 5$, Shampine and Reichelt chose an improvement in efficiency of 26% leading to $\kappa = -0.1850$ [44].
The coefficients of the MATLAB NDF methods of order 1 to 5 are given in Table 2 [1].

Table 2: Coefficients of the NDF methods of order 1 to 5.

k	κ	α_5	α_4	α_3	α_2	α_1	α_0	β_k	p	C_{p+1}	α
1	$-37/200$					1	-1	1	1	1	90°
2	$-1/9$				1	$-\frac{4}{3}$	$\frac{1}{3}$	$\frac{2}{3}$	2	$-\frac{2}{9}$	90°
3	-0.0823			1	$-\frac{18}{11}$	$\frac{9}{11}$	$-\frac{2}{11}$	$\frac{6}{11}$	3	$-\frac{3}{22}$	80°
4	-0.0415		1	$-\frac{48}{25}$	$\frac{36}{25}$	$-\frac{16}{25}$	$\frac{3}{25}$	$\frac{12}{25}$	4	$-\frac{12}{125}$	66°
5	0	1	$-\frac{300}{137}$	$\frac{300}{137}$	$-\frac{200}{137}$	$\frac{75}{137}$	$-\frac{12}{137}$	$\frac{60}{137}$	5	$-\frac{110}{137}$	51°

2.4.3 Enright's two-derivative method

In [15], Enright derived a class of second derivative multistep formulas for solving stiff equations denoted by SDMM(p). His formula is more accurate than BDFs. The

following second derivative k-step formula are considered,

$$y_{n+1} = \sum_{r=1}^{k} \alpha_r y_{n+1-r} + h \sum_{r=0}^{k} \beta_r y'_{n+1-r} + h^2 \sum_{r=0}^{k} \gamma_r y''_{n+1-r}, \tag{16}$$

where α_j, β_j and γ_j are parameters. The formula will be implicit if either β_0 or γ_0 is nonzero.

We shall introduce some particular cases of Equ. (16). Obrechkoff [35] considered the fourth order formula

$$y_{n+1} = y_n + \frac{h}{2}\left(y'_n + y'_{n+1}\right) - \frac{h^2}{12}\left(y''_n - y''_{n+1}\right), \tag{17}$$

which is A-stable but not stable at infinity. Let us recall the definition of stability at infinity.

Definition 9 *A k-step formula is stable at infinity if the following condition satisfied: There exists a negative real number W,*

$$\sup_{h\lambda<W} \left|\frac{y_n}{y_{n-1}}\right| = c < 1.$$

In [33], Liniger and Willoughby considered the following two parameter class of formulas:

$$y_{n+1} = y_n + \frac{h}{2}\left[(1-a)\, y'_n + (1+a)\, y'_{n+1}\right] + \frac{h^2}{4}\left[(b-a)y''_n - (b+a)y''_{n+1}\right]. \tag{18}$$

These methods are A-stable if a and b satisfy the conditions $\frac{1}{3} \leq a+b \leq 2$ and $0 \leq b-a \leq \frac{1}{3}$. For $a = b = \frac{1}{3}$, the method is A-stable, stable at infinity and of order three.

From Formula (16), in [15], Enright proposes a class of second derivative k-step methods of order $k+2$ up to 9 given by

$$y_{n+1} = y_n + h \sum_{r=0}^{k} \beta_r y'_{n+1-r} + h^2 \gamma_0 y''_{n+1}, \tag{19}$$

with $\alpha_i = 0$, $i = 2, 3, \ldots, k$, $\gamma_0 \neq 0$ and $\gamma_i = 0$, $i = 1, 2, \ldots, k$ these methods are stiffly stable for $1 \leq k \leq 7$ [15].

Formula (19) for $k = 1$ corresponds to the third order formula given by Liniger and Willoughby [33].

2.4.4 Ezzeddine *et al.*'s third derivative multistep methods

In [18], Ezzeddine and Hojjati construct a new class of third derivative k-step methods of order $k+3$, $k = 1, 2, \ldots, 5$, denoted by TDMM$(k+3)$. Their methods, more advanced and efficient than Enright's second derivative method and other methods [18], are defined by

$$y_{n+1} = y_n + h\sum_{r=0}^{k} \alpha_r y'_{n+1-r} + h^2\beta_k y''_{n+1} + h^3\gamma_k y'''_{n+1}. \tag{20}$$

These methods are A-stable of order $k+3$ up to order 6 and $A(\alpha)$-stable up to order 8. In order to get high accuracy and improve their absolute stability region, these methods use the first, second and third derivative of the solution.
The coefficients of the TDMM are given in Table 3.

Table 3: Coefficients of the TDMM methods of order 4 to 8.

k	α_5	α_4	α_3	α_2	α_1	α_0	γ_k	β_k
1					$\frac{3}{4}$	$\frac{1}{4}$	$\frac{1}{24}$	$-\frac{1}{4}$
2				$\frac{113}{160}$	$\frac{3}{10}$	$-\frac{1}{160}$	$\frac{7}{240}$	$-\frac{17}{80}$
3			$\frac{8813}{12960}$	$\frac{1}{3}$	$-\frac{7}{480}$	$\frac{1}{810}$	$\frac{17}{720}$	$-\frac{83}{432}$
4		$\frac{479833}{725760}$	$\frac{151}{420}$	$-\frac{41}{1680}$	$\frac{47}{11340}$	$-\frac{11}{26880}$	$\frac{41}{2016}$	$-\frac{2159}{12096}$
5	$\frac{46913609}{72576000}$	$\frac{1099}{2880}$	$-\frac{1429}{40320}$	$\frac{821}{90720}$	$-\frac{577}{322560}$	$\frac{89}{504000}$	$\frac{731}{40320}$	$-\frac{29101}{172800}$

2.4.5 Other known methods

In [10] and [11], Cash derives extended backward differentiation formulas and a family of L-stable, second derivative extended backward differentiation formula of order up to 8 and a 9th order $A(\alpha)$-stable scheme with $\alpha > 89°$. In [26], Ismail and Ibrahim construct a special class of efficient second derivative multistep methods whose stability depends on two free parameters. In [25], Hojjati, Rahimi and Hosseini present a new class of second derivative multistep methods whose $A(\alpha)$ region of stability is larger than these of several other well known methods.

Chapter 3

Three-derivative methods

In Section 3.1, we define HBO$(3,p)$, for $p = 5, 6, \ldots, 14$, and list the relevant order conditions. In Section 3.2, we consider their regions of absolute stability and in Section 3.3, their principal local truncation error coefficients. Numerical results are listed in Section 3.4.

3.1 HBO(3,p); $5 \leq p \leq 14$.

We present the implicit Hermite–Birkhoff–Obrechkoff methods of order $p = k + 4$ with $1 \leq k \leq 10$, denoted by HBO$(3,p)$. These methods use first, second and third derivatives. To integrate numerically an initial value problem as in Equ. (1) from t_n to t_{n+1}, the k-step HBO$(3,p)$ of order $p = k + 4$ is given by the formula

$$y_{n+1} = y_n + h \sum_{j=0}^{k} \beta_j y'_{n+1-j} + h^2 \left(\gamma_0 y''_{n+1} + \gamma_1 y''_n\right) + h^3 \delta_0 y'''_{n+1}. \tag{21}$$

It is to be noted that, when γ_1 in (21) equals zero, formula(21) of HBO$(3,p)$ methods and formula (20) of TDMM(p) are the same.
Using the localizing assumption for formula (21) [32], we obtain

$$\begin{aligned} y\left(t_n + h\right) = y\left(t_n\right) + h\left[\beta_0 y'\left(t_n + h\right) + \beta_1 y'\left(t_n\right) + \cdots + \beta_k y'\left(t_n - (k-1)h\right)\right] \\ + h^2\left[\gamma_0 y''\left(t_n + h\right) + \gamma_1 y''\left(t_n\right)\right] + h^3 \delta_0 y'''\left(t_n + h\right). \end{aligned} \tag{22}$$

Using a Taylor expansion of y, y', y'' and y''' at t_n, we get the following system of $p = k + 4$ linear equations in the p unknowns $\beta_j,\ 0 \le j \le k\ , \gamma_0,\ \gamma_1$ and δ_0 :

$$\begin{aligned} 1 &= \sum_{j=0}^{k} \beta_j, \\ \frac{1}{2} &= \sum_{j=0}^{k} \beta_j (1-j) + \gamma_0 + \gamma_1, \\ \frac{1}{\ell!} &= \sum_{j=0}^{k} \beta_j \frac{(1-j)^{\ell-1}}{(\ell-1)!} + \gamma_0 \frac{1}{(\ell-2)!} + \delta_0 \frac{1}{(\ell-3)!}, \qquad \ell = 3, 4, \ldots, p. \end{aligned} \tag{23}$$

The coefficient matrix of this linear system is a Vandermonde type matrix, of rank p (see for example [3], [20] and [4]). Hence, for $5 \le p \le 14$, the coefficients of the method HBO$(3, p)$ are unique.

It is to be noted that, when γ_1, in (23), is set to zero and the equation of (23) for $l = p$ is deleted, the order conditions (23) are the order conditions of TDMM(p). Numerical results stated in Subsection 3.4.3, 3.4.4 and 3.4.5 will show that HBO$(3, p)$ methods are generally more efficient than TDMM(p). For the $A(\alpha)$-stability of HBO$(3, p)$ is slightly smaller than the corresponding one of TDMM(p).

3.2 Regions of absolute stability

Recall (see Proposition 1 and Remark 1 in Section 2.2) that the region $\mathcal{R}$ of absolute stability is the subset of all points $\hat{h} \in \mathbb{C}$ such that all the roots of $\pi(r, \hat{h})$ lie inside the unit circle.
For HBO$(3, p)$, the characteristic polynomial $\pi(r, \hat{h})$ is given by $\sum_{j=0}^{k} \eta_j r^j$, where the η_j are given by:

$$\eta_k = 1, \quad d \cdot \eta_{k-1} = -\left(1 + \beta_1 \hat{h} + \gamma_1 \hat{h}^2\right),$$
$$d \cdot \eta_{k-l} = -\hat{h}\beta_l \quad 2 \le l \le k$$
$$\text{where} \quad d = 1 - \beta_0 \hat{h} - \gamma_0 \hat{h}^2 - \delta_0 \hat{h}^3.$$

Using Matlab scanning techniques, we represent the region $\mathcal{R}$ of absolute stability as

Table 4: For a given order p, the table lists the angles α of $A(\alpha)$-stability for HBO$(3,p)$, SDMM(p) and TDMM(p), respectively.

Order p	α for HBO$(3,p)$	α for SDMM(p)	α for TDMM(p)
5	90.00°	87.88°	90.00°
6	90.00°	82.03°	90.00°
7	83.66°	73.10°	89.86°
8	84.29°	59.95°	89.10°
9	83.48°	37.61°	
10	81.25°		
11	78.93°		
12	76.26°		
13	73.89°		
14	71.22°		

the exterior of the closed regions shown in Fig. 2.

Recall (Definition 3 of Chapter 2) that a method is $A(\beta)$-stable if its region of absolute stability contain $\left\{\hat{h}; -\beta < \pi - \arg\hat{h} < \beta\right\}$. For a method M, let α denote the supremum of

$$\left\{\beta \in \left(0, \frac{\pi}{2}\right); M \text{ is } A\,(\beta)\text{-stable}\right\}.$$

In Table 4, we list the values of α for HBO$(3,p)$, SDMM(p) (see Subsection 2.4.3) [24, p. 263] and TDMM(p) (see Subsection 2.4.4) [18]. From Table 4, we note that:

- the angle α is larger for HBO$(3,p)$ than for SDMM(p) for $p = 5, 6, \ldots, 9$ and decreases slowly when p increases.
- the value of the angle α is the same for HBO$(3,p)$ and TDMM(p), for order $p = 5$ and 6.
- for $p = 7$ and 8, TDMM(p) has a larger α than HBO$(3,p)$.

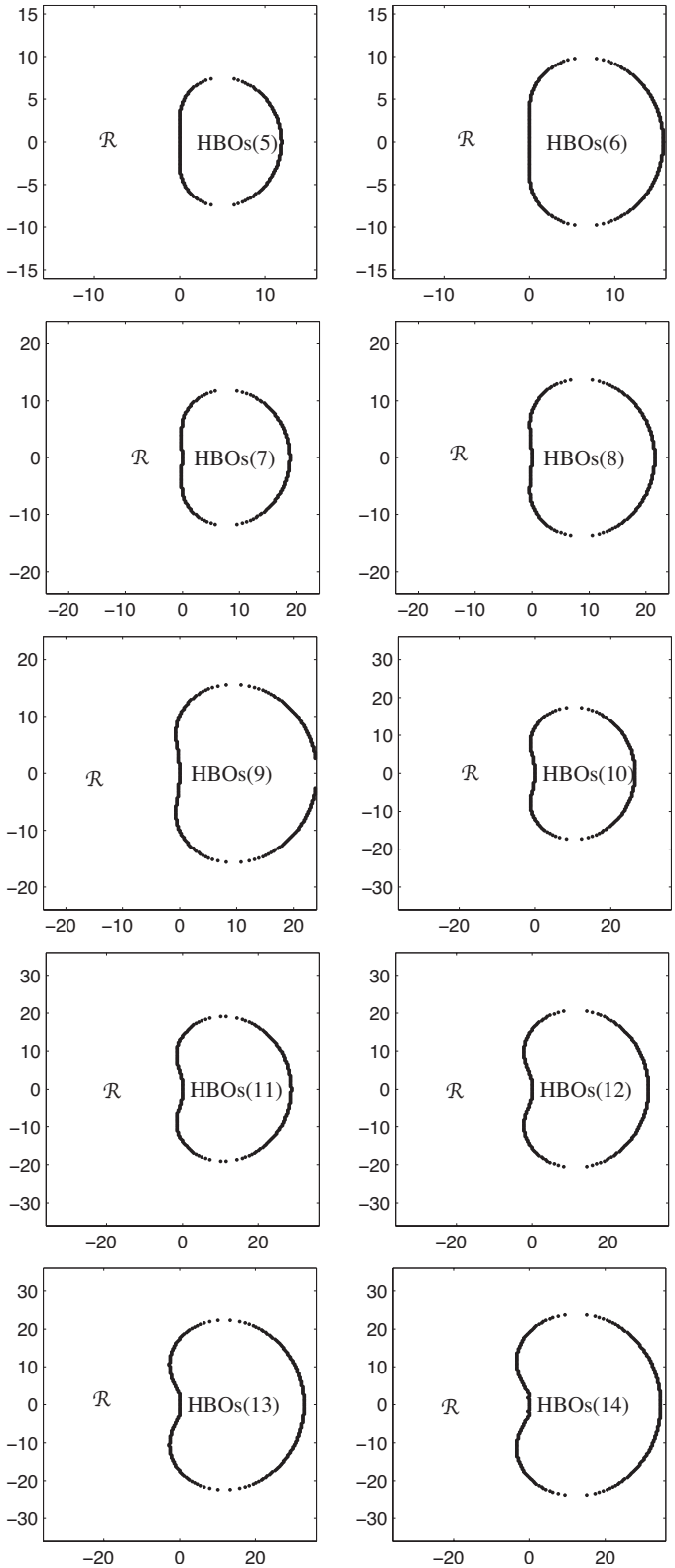


Figure 2: Regions of absolute stability, $\mathcal{R}$, of HBO(3,5–14).

Table 5: For a given order p, the table lists the PLTC of HBO$(3,p)$ and SDMM(p), respectively.

Order p	PLTC of HBO$(3,p)$	PLTC of SDMM(p)
5	-1.39e-04	2.36e-03
6	-3.31e-05	1.36e-03
7	-1.16e-05	8.63e-04
8	-5.01e-06	5.90e-04
9	-2.49e-06	4.24e-04
10	-1.36e-06	
11	-8.04e-07	
12	-5.01e-07	
13	-3.28e-07	
14	-2.22e-07	

3.3 Principal error term

Recall that as in [32], the principal error term is obtained by the Taylor expansion of Formula (21) and it is given by

$$\left\{\frac{1}{(p+1)!}-\left[\sum_{j=0}^{k}\beta_j\frac{(1-j)^p}{p!}+\gamma_0\frac{(1)^{(p-1)}}{(p-1)!}+\delta_0\frac{1}{(p-2)!}\right]\right\}h^{p+1}y_n^{(p+1)}. \tag{24}$$

In Table 5, we list the principal local truncation error coefficients (PLTC), which are equal to

$$\left\{\frac{1}{(p+1)!}-\left[\sum_{j=0}^{k}\beta_j\frac{(1-j)^p}{p!}+\gamma_0\frac{(1)^{(p-1)}}{(p-1)!}+\delta_0\frac{1}{(p-2)!}\right]\right\}$$

for HBO$(3,p)$ and SDMM(p) (see [24, p. 263]). We note that HBO$(3,p)$ has smaller PLTC than SDMM(p) for all p.

3.4 Numerical Results

3.4.1 Iteration scheme

The implicit equations of Formula (21) are implemented with constant steps and solved iteratively by the modified Newton–Raphson method [32, p. 13]. This method is one of the most powerful techniques for solving numerically systems of linear equations. The modified Newton–Raphson takes the following form:

$$J_{n+1}^0\left(y_{n+1}^{l+1}-y_{n+1}^l\right)=-y_{n+1}^l+h\beta_0 f(t_{n+1},y_{n+1}^l)+h^2\gamma_0\frac{d}{dt}f(t_{n+1},y_{n+1}^l)$$
$$+h^3\delta_0\frac{d^2}{dt^2}f(t_{n+1},y_{n+1}^l)+y_n+h\sum_{j=1}^{k}\beta_j y'_{n+1-j}+h^2\gamma_1 y''_n,\quad l=0,1,2,\ldots, \tag{25}$$

where the Jacobian J_{n+1}^0 at step $l=0$, is given by

$$J_{n+1}^0=\left[I-h\beta_0\frac{\partial f(t_{n+1},y_{n+1}^0)}{\partial y}-h^2\gamma_0\frac{\partial\frac{d}{dt}f(t_{n+1},y_{n+1}^0)}{\partial y}-h^3\delta_0\frac{\partial\frac{d^2}{dt^2}f(t_{n+1},y_{n+1}^0)}{\partial y}\right]. \tag{26}$$

We do not need to update the Jacobian J_{n+1}^0 for the modified Newton–Raphson method to get good results.

As a first guess, we use

$$y_{n+1}^0=y_{n+1}^P, \tag{27}$$

obtained from the following predictor, similar to Gear predictors [21],

$$y_{n+1}^P=\sum_{j=1}^{k}\alpha_{P,j}y_{n+1-j}+h\beta_P y'_n. \tag{28}$$

At each iteration of the modified Newton–Raphson method, the derivatives $y''_{n+1}=\frac{d}{dt}f(t_{n+1},y_{n+1})$ and $y'''_{n+1}=\frac{d^2}{dt^2}f(t_{n+1},y_{n+1})$ of the Taylor series are calculated with known recurrence formulas (see, for example, [23, pp. 46–49], [5]). In the rest of this chapter, we compare numerically our new methods with BDF(5), SDMM(9), TDMM(9), TDMM(11) and TDMM(13).

Recall that the infinity norm or uniform norm of a vector $\nu\in\mathbb{R}^n$ is defined by

$$\|\nu\|_\infty=\max_{1\le i\le n}|\nu_i|,$$

and the *error at the endpoint of the integration interval*, denoted by EPE, is equal to

$$\text{EPE} = \|y_{\text{end}} - z_{\text{end}}\|_\infty,$$

where y_{end} is the numerical value obtained by the numerical method at the endpoint t_{end} of the integration interval and z_{end} is the "exact solution". This "exact solution" is obtained by MATLAB's `ode15s` with one of the most stringent tolerances: 5×10^{-14}. Recall that MATLAB's `ode15s` is a variable order solver based on the numerical differentiation formulas (NDFs). Optionally, it uses the backward differentiation formulas (BDFs, also known as Gear's method) that are usually less efficient [44].

3.4.2 Implementation and problems used for comparison

We compare the numerical performances of HBO$(3, p)$, BDF(5), SDMM(9) and TDMM(p) on each of the test problems listed below:

- **vdP,** the van der Pol oscillator Equ. (29).
- **RC,** the Robertson chemical reaction Equ. (32).
- **D1,** the Stiff DETEST problem D1 Equ. (33).

Our comparison of the performance of the methods proceeds in five steps.
Firstly, we implement HBO$(3, p)$, BDF(5), SDMM(9) and TDMM(p) in MATLAB.
Secondly, we collect the CPU time and the endpoint error.
Thirdly, we compute the CPU percentage efficiency gain (the estimates of the CPU time are obtained by using MATLAB's polyfit).
Fourthly, we compute the estimates of number of steps, denoted by NST (the estimates of NST are obtained by using MATLAB's polyfit).
Finally, we draw figures and tables according to the collected data and compare the results.

The necessary starting values at $t_1, t_2, \ldots, t_{k-1}$ for HBO$(3, p)$ were obtained by MATLAB's `ode15s` with stringent tolerance 5×10^{-14}. Computations were performed in MATLAB and C++ on a PC with the following characteristics: Memory: 5.8 GB, Processor 0,1,...,7: Intel(R) Core(TM) i7 CPU 920 @ 2.67GHz, Operating system: Ubuntu Release 11.04, Kernel Linux 2.6.38-12-generic, GENOME 2.32.1.

3.4.3 Comparing CPU time on the van der Pol oscillator

Van der Pol oscillator was originally proposed by the Dutch electrical engineer and physicist Balthasar Van der Pol while he was working at Philips Company in Eindhoven [7]. This model is important for oscillatory processes not only in physics, but also in biology, sociology and even economics [31].
The van der Pol oscillator is given by the following equation [25]:

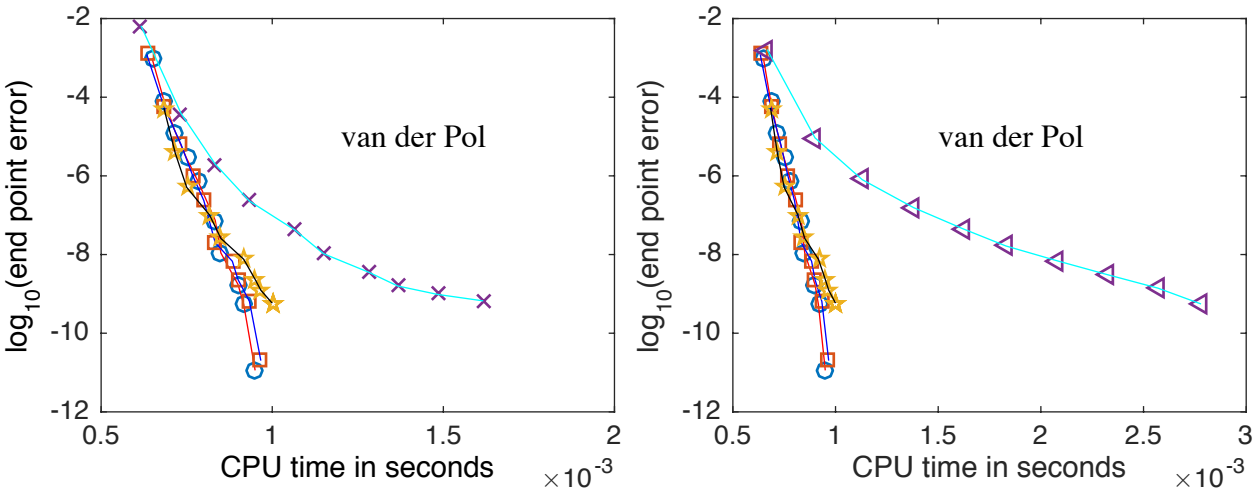
Problem 1 (vdP) *The van der Pol oscillator is the system*

$$\begin{aligned} y_1' &= y_2, & y_1(0) &= 2, \\ y_2' &= \left[\left(1 - y_1^2\right) y_2 - y_1\right] \mu^2, & y_2(0) &= 0, \end{aligned} \tag{29}$$

with $\mu = 500$.

In our comparison of methods, we will use the interval $[0, t_{end}] = [0, 0.8]$. Notice that this choice is arbitrary, and we could have used larger t_{end} (for example $t_{end} \geq 11$). We use this system to compare HBO$(3, p)$ with BDF(5), SDMM(9) and TDMM(p). In Figure 3, we draw $\log_{10}(\text{EPE})$ as a function of CPU time for HBO$(3, p)$, $p = 9, 11, 13$, SDMM(9) and BDF(5). From this figure, we deduce that for $p = 9, 11, 13$, HBO$(3, p)$ compare favorably with BDF(5) and SDMM(9) on Problem (vdP) at stringent tolerances.
We note that the results for HBO$(3, p)$, $p = 9, 11, 13$, are very close to each other.

In [42], Sharp introduces the CPU percentage efficiency gain (CPU PEG). To compute CPU PEG, we need the following intermediate steps, given a whole set of data $(\log_{10}(\text{EPE}), \log_{10}(\text{CPU}))$ obtained by MATLAB.



HBO(3,13) ∘, HBO(3,11) □, HBO(3,9) ⋆, SDMM(9) × and BDF(5) ◁

Figure 3: HBO$(3,p)$, $p = 9, 11, 13$, are compared with SDMM(9) (left) and BDF(5) (right) for Problem (vdP)

1. Using these data $(\log_{10}(\text{EPE}), \log_{10}(\text{CPU}))$, we consider a function F_i whose graph approximates well our set of data.

2. Using Polyval, we get the max of error in estimation and the estimate of the CPU time.

3. We compute CPU PEG defined by the Formula (30).

The *CPU percentage efficiency gain* (CPU PEG) is defined by the formula (cf. Sharp [42]),

$$(\text{CPU PEG})_i = 100\left[\frac{\sum_j \text{CPU}_{2,i,j}}{\sum_j \text{CPU}_{1,i,j}} - 1\right], \tag{30}$$

where $\text{CPU}_{1,i,j}$ and $\text{CPU}_{2,i,j}$ are the estimate of CPU time of methods 1 and 2, respectively, associated with problem i, and the estimate of EPE $= 10^{-j}$. To compute $\text{CPU}_{2,i,j}$ and $\text{CPU}_{1,i,j}$ appearing in (30), we obtain a straight line L that best fits the data $(\log_{10}(\text{EPE}), \log_{10}(\text{CPU}))$ in a least-squares sense by MATLAB's `polyfit`. (For example, Figure 4 shows the straight line that best fits the data $(\log_{10}(\text{EPE}), \log_{10}(\text{CPU}))$ of HBO(3,9) solving Problem (vdP).) Then, for chosen integer values of the summation index j, we take the estimate of EPE $= 10^{-j}$ and obtain $\log_{10}$(the estimate of CPU time)

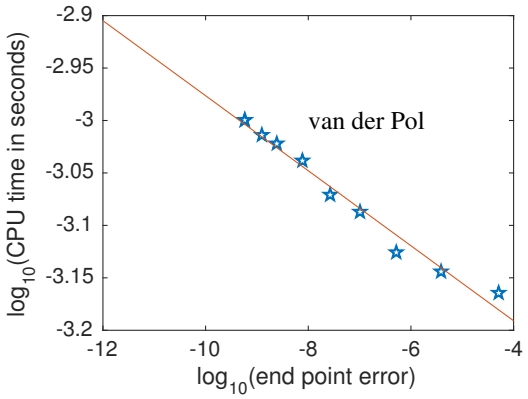


Figure 4: straight line that best fits the data $(\log_{10}(\text{EPE}), \log_{10}(\text{CPU}))$ of HBO(3,9) solving Problem (vdP)

from the approximating line L, and finally the estimate of CPU.

The *number-of-steps percentage efficiency gain* (NST PEG) is defined by the formula (cf. Sharp [42]),

$$(\text{NST PEG})_i = 100\left[\frac{\sum_j \text{NST}_{2,i,j}}{\sum_j \text{NST}_{1,i,j}} - 1\right], \tag{31}$$

where $\text{NST}_{1,i,j}$ and $\text{NST}_{2,i,j}$ denote the estimate of the number of steps of methods 1 and 2, respectively, associated with problem i, and the estimate of EPE $= 10^{-j}$. The number of steps was obtained from the line which fit the data $(\log_{10}(\text{EPE}), \log_{10}(\text{NST}))$ in a least-squares sense by means of MATLAB's `polyfit`.

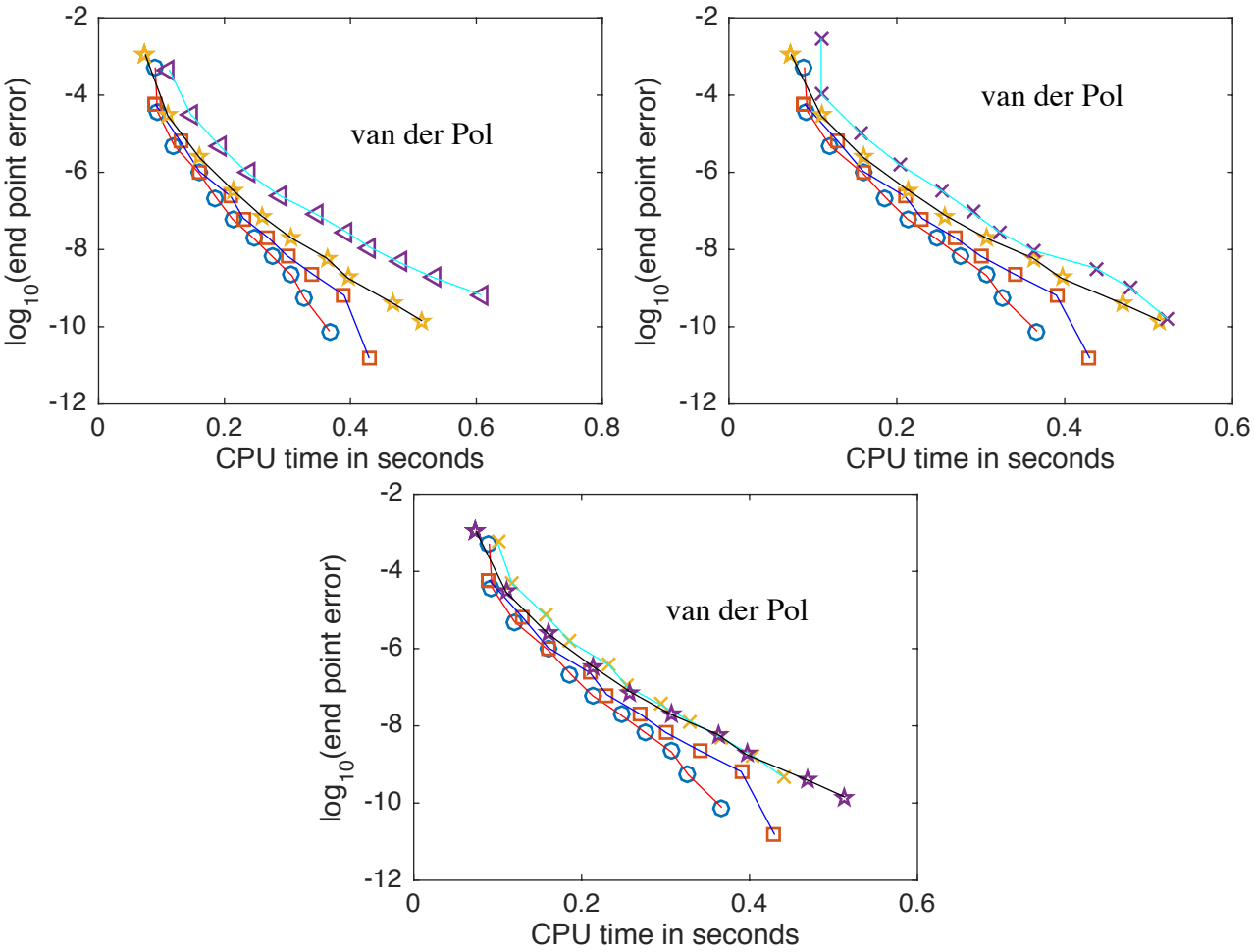
Table 6 lists the CPU PEG and NST PEG, defined by Formulas (30) and (31), respectively, of HBO$(3,p)$, $p = 9, 11, 13$, over SDMM(9) and BDF(5) for Problem (vdP). From these results, we can conclude that HBO$(3,p)$, $p = 9, 11, 13$, generally need less CPU time than SDMM(9) and BDF(5) .

Comparing CPU time of HBO$(3,p)$ and TDMM(p) on the van der Pol oscillator

In Figure 5, we plot $\log_{10}(\text{EPE})$ as a function of CPU time in seconds for HBO$(3,p)$ and TDMM(p), for $p = 9, 11, 13$. From this figure, we derive that for $p = 9, 11, 13$ HBO$(3,p)$ compare favorably with TDMM(p) for Problem (vdP) at stringent tolerances.

Table 6: CPU PEG and NST PEG of the listed HBO$(3,p)$ over SDMM(9) and BDF(5) for Problem (vdP).

	CPU PEG of listed HBO over:		NST PEG of listed HBO over:	
HBO$(3,p)$	SDMM(9)	BDF(5)	SDMM(9)	BDF(5)
HBO(3,9)	25%	69%	79%	858%
HBO(3,11)	32%	100%	141%	1332%
HBO(3,13)	32%	101%	174%	1526%



HBO(3,13) ∘, HBO(3,11) □, HBO(3,9) ⋆
TDMM(9) ◁, TDMM(11) × and TDMM(13) ×

Figure 5: HBO$(3,p)$, $p = 9, 11, 13$, are compared with TDMM(9) (top left), TDMM(11) (top right) and TDMM(13) (bottom) for Problem (vdP).

Table 7: CPU PEG and NST PEG of the listed HBO$(3,p)$ over TDMM(p) for Problem (vdP).

	CPU PEG of listed HBO over:			NST PEG of listed HBO over:		
HBO$(3,p)$	TDMM(9)	TDMM(11)	TDMM(13)	TDMM(9)	TDMM(11)	TDMM(13)
HBO(3,9)	37%	13%	4%	37%	11%	1%
HBO(3,11)	62%	34%	24%	67%	36%	23%
HBO(3,13)	73%	44%	33%	80%	46%	33%

Table 7 lists the CPU PEG and NST PEG, defined by Formulas (30) and (31), respectively, of HBO$(3,p)$ over TDMM(p), for $p = 9, 11, 13$ for Problem (vdP). From the results, we can conclude that HBO$(3,p)$, $p = 9, 11, 13$, often use less CPU time than other methods.

3.4.4 Comparing CPU time on the Robertson chemical reaction problem

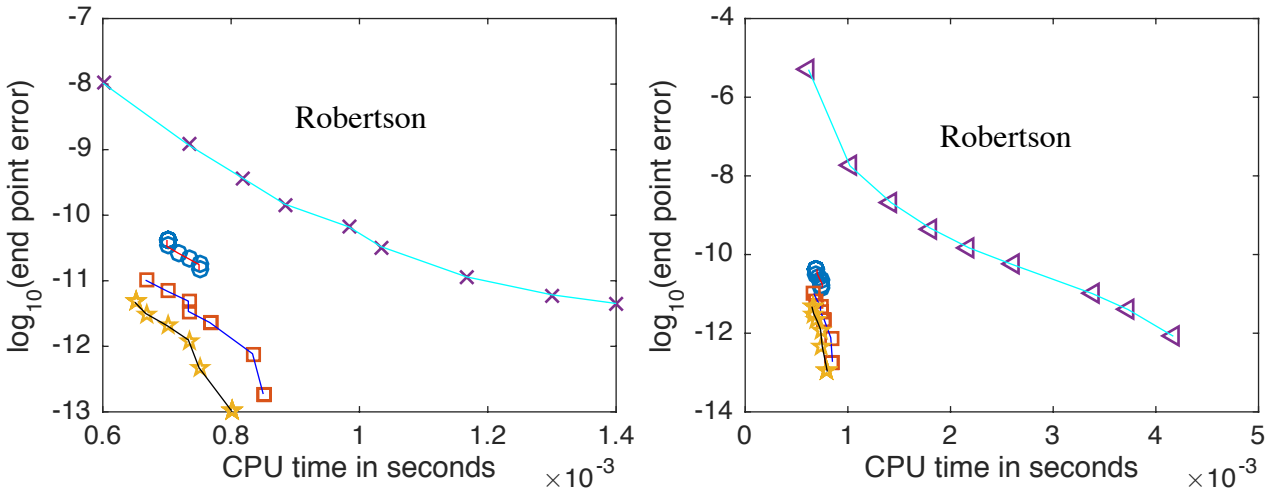
As a second comparison between HBO$(3,p)$, BDF(5), SDMM(9) and TDMM(p), we consider the Robertson chemical reaction which was studied by H.H. Robertson in 1966 [39, pp. 178–182]. This problem is one of the most studied ordinary differential equations in numerical studies [14] and is usually used as a test problem to compare between different stiff integrators.

Problem 2 (RC) *Robertson chemical reaction:*

$$\begin{aligned} y_1' &= -0.04y_1 + 10^4 y_2 y_3, & y_1(0) &= 1, \\ y_2' &= 0.04y_1 - 10^4 y_2 y_3 - 3\times 10^7 y_2^2, & y_2(0) &= 0, \\ y_3' &= 3\times 10^7 y_2^2, & y_3(0) &= 0, \end{aligned} \tag{32}$$

with $t_{end} = 400$.

In Figure 6, we draw $\log_{10}(\text{EPE})$ (vertical axis) as a function of CPU time for HBO$(3,p)$, $p = 9, 11, 13$, SDMM(9) and BDF(5). From this figure, we conclude that HBO$(3,p)$, $p = 9, 11, 13$, compare favorably with BDF(5) and SDMM(9) on Problem (RC) at stringent tolerances. Surprisingly here, HBO(3,9) is more efficient



HBO(3,9) ⋆, HBO(3,11) □, HBO(3,13) ∘, SDMM(9) ×, and BDF(5) ◁

Figure 6: HBO(3,9), HBO(3,11) and HBO(3,13) are compared with SDMM(9) (left), and BDF(5) (right) for Problem (RC).

than HBO$(3,p)$, $p = 11, 13$.

Table 8 lists the CPU PEG and the NST PEG, defined by Formulas (30) and (31), respectively, of HBO$(3,p)$, $p = 9, 11, 13$, over SDMM(9) and BDF(5) for Problem (RC). From these results, we can conclude that HBO$(3,p)$, $p = 9, 11, 13$, generally require less CPU time than SDMM(9) and BDF(5).

As an example, HBO(3,9) uses a step size of length 10 and takes 40 steps, compared to 1167 steps used by SDMM(9) and 7000 steps used by BDF(5) to obtain an EPE of about 4.0e-12.

Table 8: CPU PEG and NST PEG of HBO$(3,p)$, $p = 9, 11, 13$ methods over SDMM(9) and BDF(5) for Problem (RC).

	CPU PEG of HBO$(3,p)$ over:		NST PEG of HBO$(3,p)$over:	
HBO$(3,p)$	SDMM(9)	BDF(5)	SDMM(9)	BDF(5)
HBO(3,9)	120%	507%	6 269%	36 274%
HBO(3,11)	99%	450%	5 147%	29 869%
HBO(3,13)	51%	280%	713%	6 961%

Table 9: CPU PEG and NST PEG of the listed HBO$(3,p)$ over TDMM(p) for Problem (RC).

	CPU PEG of listed HBO over:			NST PEG of listed HBO over:		
HBO$(3,p)$	TDMM(9)	TDMM(11)	TDMM(13)	TDMM(9)	TDMM(11)	TDMM(13)
HBO(3,9)	519%	683%	38%	466%	640%	35%
HBO(3,11)	376%	502%	6%	368%	512%	12%
HBO(3,13)	-12%	9%	-79%	-17%	6%	-79%

Comparing CPU time of HBO$(3,p)$ and TDMM(p) on the Robertson chemical reaction problem

In Figure 7, we plot $\log_{10}(\text{EPE})$ as a function of the CPU time for HBO$(3,p)$ and TDMM(p), for $p = 9, 11, 13$. From this figure, we deduce that HBO$(3,p)$ compare favorably with TDMM(p), for $p = 9, 11, 13$ on Problem (RC) at stringent tolerances.

Table 9 lists the CPU PEG and the NST PEG, defined by Formulas (30) and (31), respectively, of HBO$(3,p)$ over TDMM(p), for $p = 9, 11, 13$ for Problem (RC). From the results, we can conclude that in most cases HBO$(3,p)$ requires less CPU time than TDMM(p), for $p = 9, 11, 13$.

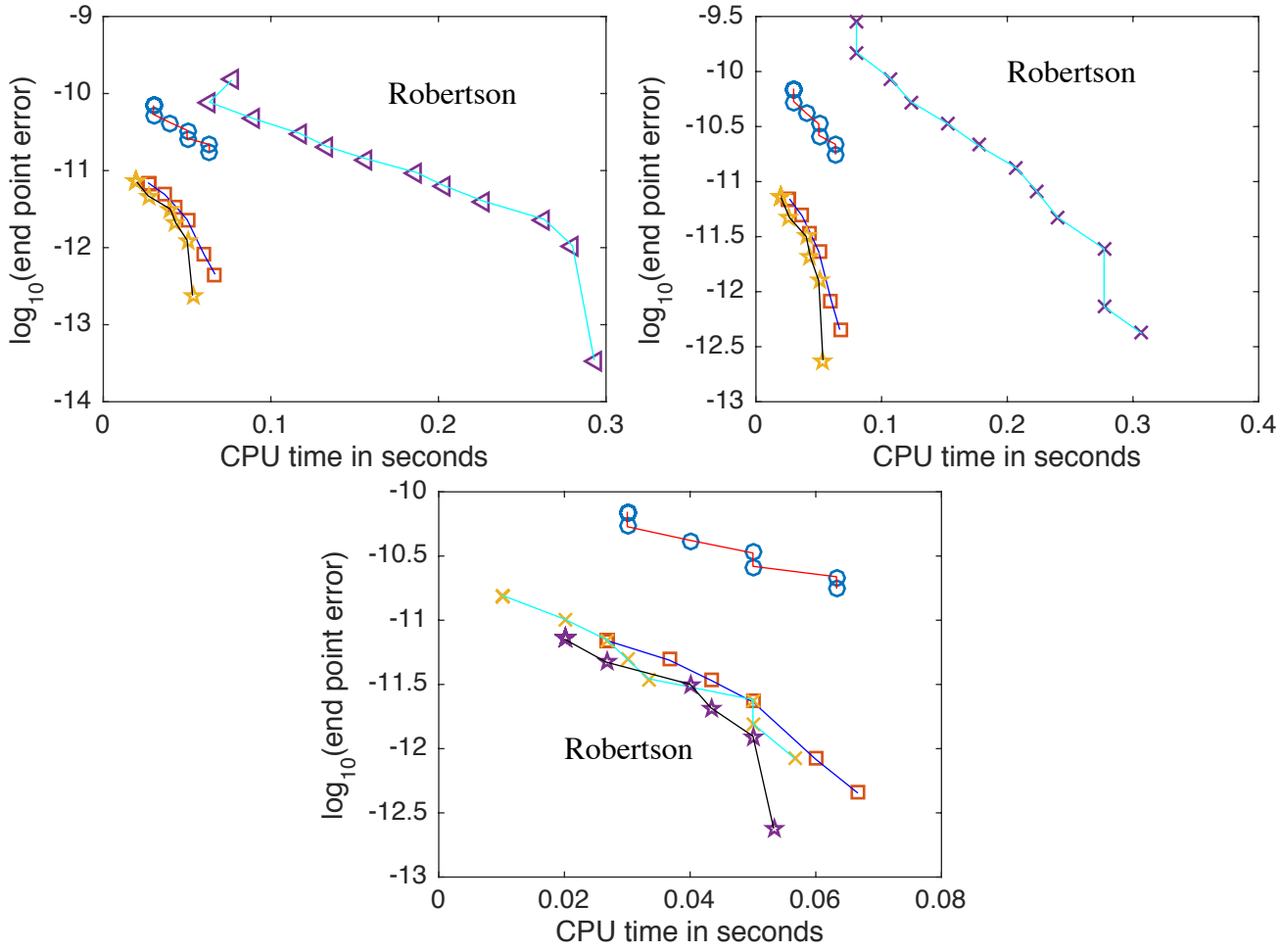
3.4.5 Comparing CPU time on a stiff DETEST problem

In 1975, Enright, Hull and Lindberg [16] presented a bank of test problems called stiff DETEST for comparing different numerical ODE solvers. This bank of problems includes 25 stiff problems, and they have been used by several authors (see for example [37, 29, 6]). As a third comparison, we consider the stiff DETEST problem D1.

Problem 3 (D1) *Stiff DETEST problem D1 [17]:*

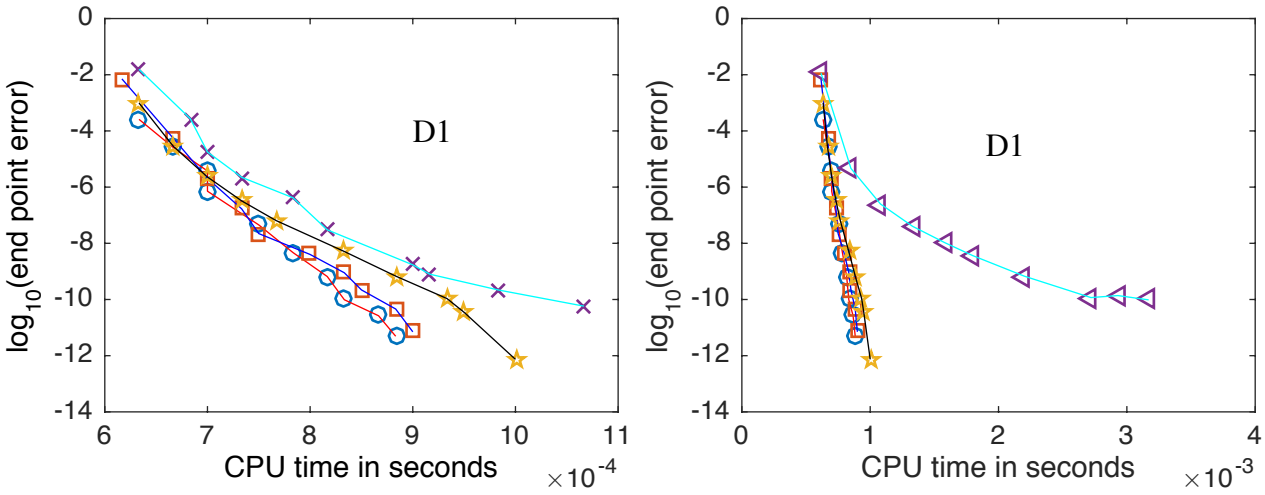
$$\begin{aligned} y_1' &= 0.2\,(y_2 - y_1)\,, & y_1(0) &= 0, \\ y_2' &= 10y_1 - (60 - 0.123y_3)\,y_2 + 0.125y_3, & y_2(0) &= 0, \\ y_3' &= 1, & y_3(0) &= 0, \end{aligned} \tag{33}$$

with $t_{end} = 400$.



HBO(3,9) ⋆, HBO(3,11) □, HBO(3,13) ∘
TDMM(9) ◁, TDMM(11) × and TDMM(13) ×

Figure 7: HBO(3,9), HBO(3,11) and HBO(3,13) are compared with TDMM(9) (top left), TDMM(11) (top right) and TDMM(13) (bottom) for Problem (RC).



HBO(3,13) ∘, HBO(3,11) □, HBO(3,9) ⋆, SDMM(9) × , and BDF(5) ◁

Figure 8: HBO$(3,p)$, $p = 9, 11, 13$, are compared with SDMM(9) (left) and BDF(5) (right) for Problem (D1).

In Figure 8, we plot $\log_{10}$(EPE) as a function of the CPU time for HBO$(3,p)$, $p = 9, 11, 13$, BDF(5) and SDMM(9). From this figure, we derive that for $p = 9, 11, 13$, HBO$(3,p)$ compares favorably with BDF(5) and SDMM(9) for Problem (D1) at stringent tolerances.

Table 10 lists the CPU PEG and the NST PEG, defined by Formulas (30) and (31), respectively, of HBO$(3,p)$, $p = 9, 11, 13$, over SDMM(9) and BDF(5) for Problem (D1). From the results, we can conclude that for $p = 9, 11, 13$, HBO$(3,p)$ generally need less CPU time than BDF(5) and SDMM(9).

Table 10: CPU PEG and NST PEG of HBO$(3,p)$, $p = 9, 11, 13$, over BDF(5) and SDMM(9) for Problem (D1).

	CPU PEG of listed HBO over:		NST PEG of listed HBO over:	
HBO$(3,p)$	SDMM(9)	BDF(5)	SDMM(9)	BDF(5)
HBO(3,9)	7%	108%	114%	2020%
HBO(3,11)	10%	114%	149%	2 508%
HBO(3,13)	12%	118%	199%	3 037%

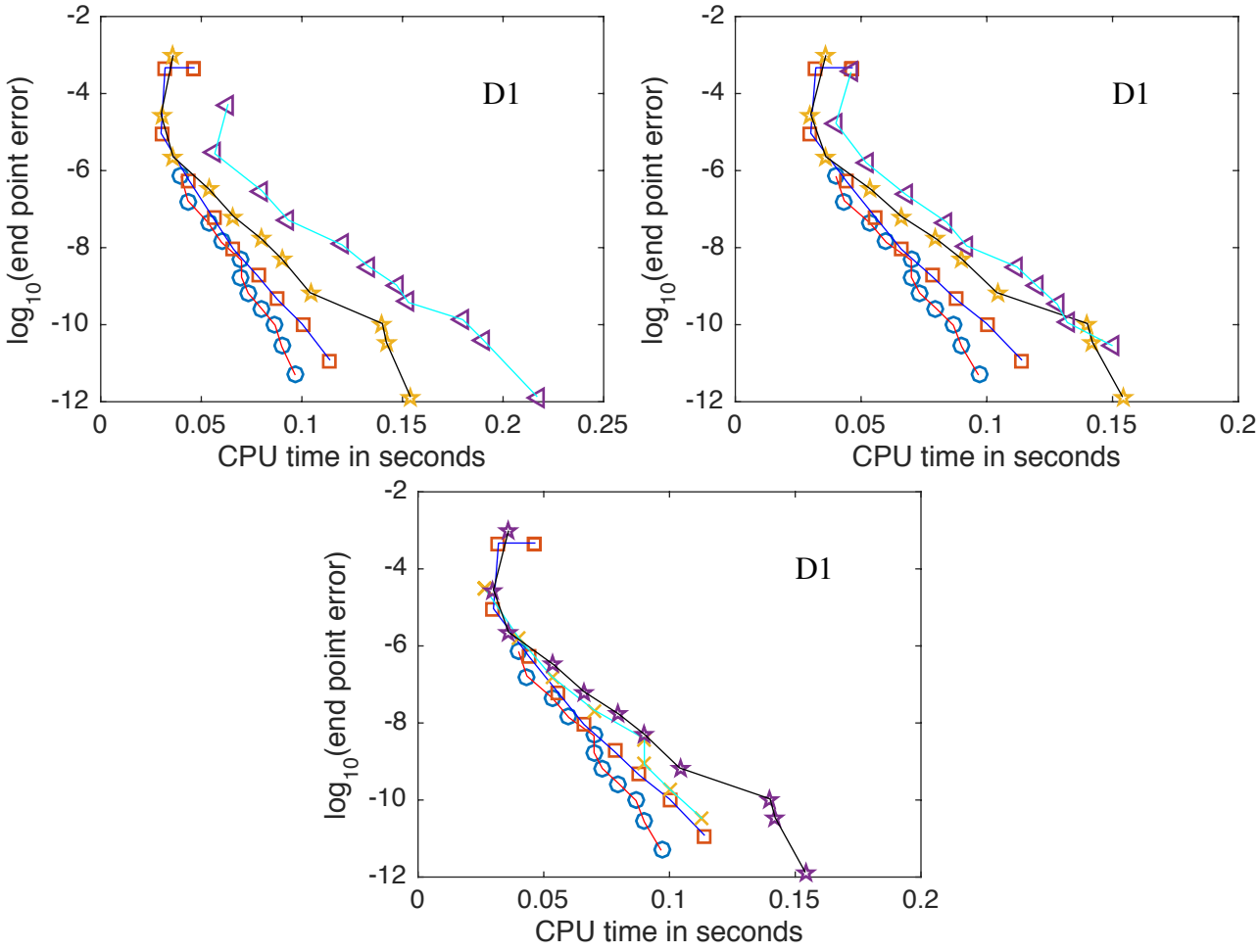
Table 11: CPU PEG and NST PEG of the listed HBO$(3,p)$ over TDMM(p) for Problem (D1).

	CPU PEG of listed HBO over:			NST PEG of listed HBO over:		
HBO$(3,p)$	TDMM(9)	TDMM(11)	TDMM(13)	TDMM(9)	TDMM(11)	TDMM(13)
HBO(3,9)	45%	19%	-12%	36%	7%	-8%
HBO(3,11)	61%	30%	-4%	69%	33%	12%
HBO(3,13)	92%	55%	12%	87%	47%	24%

Comparing CPU time of HBO(3, p) and TDMM(p) on the stiff DETEST problem D1

In Figure 9, we plot for $p = 9, 11, 13$, $\log_{10}(\text{EPE})$ as a function of the CPU time for HBO$(3,p)$ and TDMM(p). From this figure, we conclude that for $p = 9, 11, 13$, HBO$(3,p)$ compare favorably with TDMM(p) for Problem (D1) at stringent tolerances.

Table 11 lists for Problem (D1) the CPU PEG and the NST PEG of HBO$(3,p)$, over TDMM(p), for $p = 9, 11, 13$. From the results, we conclude that in most cases HBO$(3,p)$ need less CPU time than TDMM(p), for $p = 9, 11, 13$.



HBO(3,13) ∘, HBO(3,11) □, HBO(3,9) ⋆
TDMM(9) ◁, TDMM(11) × and TDMM(13) ×

Figure 9: HBO$(3,p)$, $p = 9, 11, 13$, are compared with TDMM(9) (top left), TDMM(11) (top right) and TDMM(13) (bottom) for Problem (D1).

Chapter 4

Four-derivative methods

In Section 4.1, we define HBO$(4,p)$, for $p = 7, 8, \ldots, 14$, and list the relevant order conditions. In Section 4.2, we consider their regions of absolute stability and in Section 4.3, their principal local truncation error coefficients. Numerical results are listed in Section 4.4.

4.1 HBO(4,p); $7 \leq p \leq 14$.

We present the implicit Hermite–Birkhoff–Obrechkoff methods of order $p = k + 6$ with $1 \leq k \leq 8$, denoted by HBO$(4,p)$. These methods use first, second, third and fourth derivatives. To integrate numerically an initial value problem as in Equ. (1) from t_n to t_{n+1}, the k-step HBO$(4,p)$ of order $p = k + 6$ is given by the formula

$$y_{n+1} = y_n + h \sum_{j=0}^{k} \beta_j y'_{n+1-j} + h^2 \sum_{j=0}^{1} \gamma_j y''_{n+1-j} + h^3 \sum_{j=0}^{1} \delta_j y'''_{n+1-j} + h^4 \eta_0 y^{(4)}_{n+1}. \qquad (34)$$

In order to obtain order conditions, we use the same procedure as mentioned in (Chapter 3, Section 3.1). The following order conditions are obtained from (34),

$$\begin{aligned}
1 &= \sum_{j=0}^{k} \beta_j, \\
\frac{1}{2!} &= \sum_{j=0}^{k} \beta_j (1-j) + \gamma_0 + \gamma_1, \\
\frac{1}{3!} &= \sum_{j=0}^{k} \beta_j \frac{(1-j)^2}{2!} + \sum_{j=0}^{1} \gamma_j (1-j) + \delta_0 + \delta_1, \\
\frac{1}{\ell!} &= \sum_{j=0}^{k} \beta_j \frac{(1-j)^{\ell-1}}{(\ell-1)!} + \gamma_0 \frac{1}{(\ell-2)!} \\
&\quad + \delta_0 \frac{1}{(\ell-3)!} + \eta_0 \frac{1}{(\ell-4)!}, \qquad \ell = 4, 5, \ldots, p.
\end{aligned} \tag{35}$$

These conditions form a system of $p = k + 6$ linear equations in p unknowns. As in Section 3.1, we can check that as this system has a unique solution, the coefficients of the method HBO$(4, p)$ are unique.

4.2 Regions of absolute stability

Recall (see Proposition 1 and Remark 1 in Section 2.2) that the region $\mathcal{R}$ of absolute stability is the subset of all points $\hat{h} \in \mathbb{C}$ such that all the roots of $\pi(r, \hat{h})$ lie inside the unit circle.

For HBO$(4, p)$, the characteristic polynomial $\pi(r, \hat{h})$ is given by $\sum_{j=0}^{k} \mu_j r^j$, where the μ_j are given by the following :

$$\mu_k = 1, \quad d \cdot \mu_{k-1} = -\left(1 + \beta_1 \hat{h} + \gamma_1 \hat{h}^2 + \delta_1 \hat{h}^3\right),$$

$$d \cdot \mu_{k-l} = -\hat{h}\beta_l \quad 2 \leq l \leq k$$

$$\text{where} \quad d = 1 - \beta_0 \hat{h} - \gamma_0 \hat{h}^2 - \delta_0 \hat{h}^3 - \hat{h}^4 \eta_0.$$

Using Matlab scanning techniques, we represent the region $\mathcal{R}$ of absolute stability as

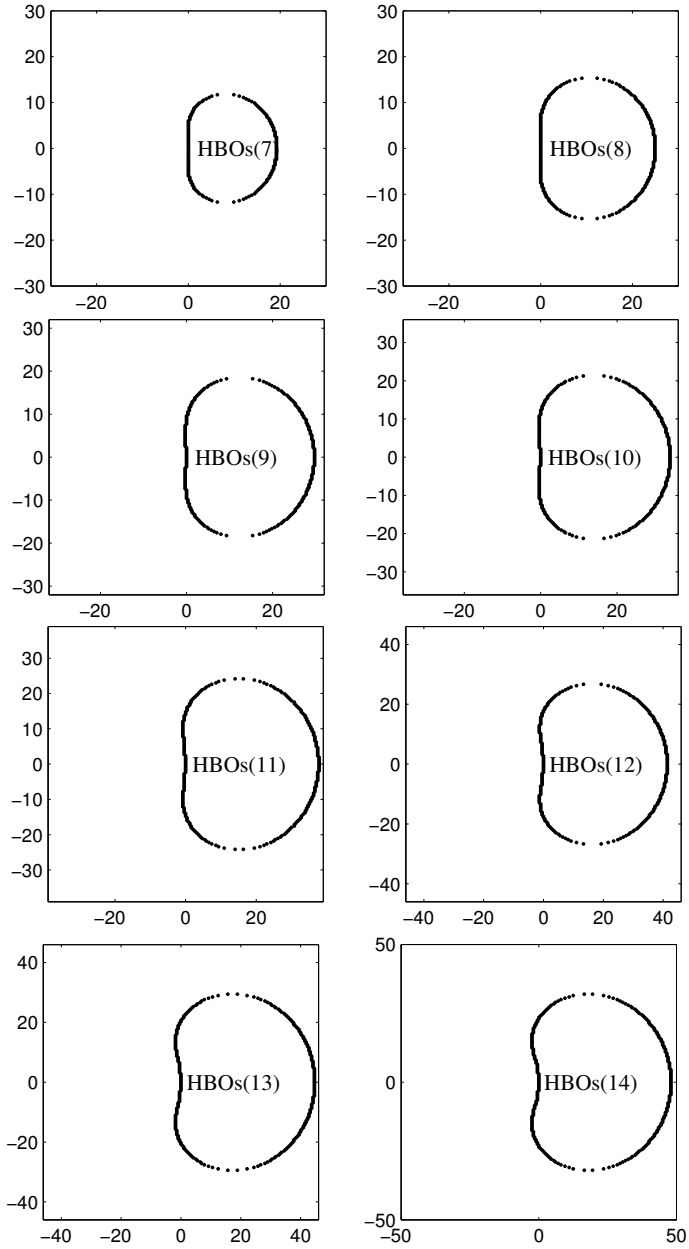


Figure 10: Regions of absolute stability, $\mathcal{R}$, of HBO(4,7–14).

Table 12: For a given order p, the table lists the angles α of $A(\alpha)$-stability for HBO$(4,p)$ and SDMM(p), respectively.

Order p	α for HBO$(4,p)$	α for SDMM(p)	α for TDMM(p)
7	90.00°	73.10°	89.86°
8	90.00°	59.95°	89.10°
9	82.87°	37.61°	
10	81.87°		
11	81.87°		
12	81.87°		
13	80.54°		
14	78.69°		

the exterior of the closed regions shown in Fig. 10.

Table 12 lists the angles α of $A(\alpha)$-stability of HBO(4,7–14), SDMM(7–9) and TDMM(7–8) [24, p. 263] and [18], respectively. From Table 12, we note that:

- i): the angle α is larger for HBO$(4,p)$ than for SDMM(p) for $p = 7, 8,$ and 9.
- ii): for $p = 7$ and 8, HBO$(4,p)$ has a larger α than TDMM(p).

4.3 The principal error term

Recall that as in [32] the principal error term is obtained by the Taylor expansion of Formula (34) and it is given by

$$\left\{\frac{1}{(p+1)!}-\left[\sum_{j=0}^{k}\beta_j\frac{(1-j)^p}{p!}+\sum_{j=0}^{1}\gamma_j\frac{(1-j)^{(p-1)}}{(p-1)!}+\sum_{j=0}^{1}\delta_j\frac{(1-j)^{(p-2)}}{(p-2)!}+\eta_0\frac{1}{(p-3)!}\right]\right\}h^{p+1}y_n^{(p+1)}. \tag{36}$$

In Table 13, we list the principal local truncation error coefficients (PLTC), which are equal to

$$\left\{\frac{1}{(p+1)!}-\left[\sum_{j=0}^{k}\beta_j\frac{(1-j)^p}{p!}+\sum_{j=0}^{1}\gamma_j\frac{(1-j)^{(p-1)}}{(p-1)!}+\sum_{j=0}^{1}\delta_j\frac{(1-j)^{(p-2)}}{(p-2)!}+\eta_0\frac{1}{(p-3)!}\right]\right\}$$

Table 13: For a given order p, the table lists the PLTC of HBO$(4,p)$ and SDMM(p), respectively.

Order p	PLTC of HBO$(4,p)$	PLTC of SDMM(p)
7	7.09e-07	8.63e-04
8	1.28e-07	5.90e-04
9	3.50e-08	4.24e-04
10	1.21e-08	
11	4.95e-09	
12	2.26e-09	
13	1.13e-09	
14	6.04e-10	

for HBO$(4,p)$ and SDMM(p).

Table 13 lists the principal local truncation error coefficients (PLTC) of HBO$(4,p)$ and SDMM(p) (see [24, p. 263]). We remark that HBO$(4,p)$ has smaller PLTC than SDMM(p) for all p.

4.4 Numerical Results

4.4.1 Iteration scheme

The implicit equations of Formula (34) are implemented with constant steps and solved iteratively by the modified Newton–Raphson method [32, p. 13]:

$$\begin{aligned} J^0_{n+1}\left(y^{l+1}_{n+1}-y^l_{n+1}\right) &= -y^l_{n+1}+h\beta_0 f(t_{n+1},y^l_{n+1})+h^2\gamma_0\frac{d}{dt}f(t_{n+1},y^l_{n+1}) \\ &\quad + h^3\delta_0\frac{d^2}{dt^2}f(t_{n+1},y^l_{n+1})+h^4\eta_0\frac{d^3}{dt^3}f(t_{n+1},y^l_{n+1}) \\ &\quad + y_n+h\sum_{j=1}^{k}\beta_j y'_{n+1-j}+h^2\gamma_1 y''_n+h^3\delta_1 y'''_n, \quad l=0,1,2,\ldots, \end{aligned} \tag{37}$$

where the Jacobian J^0_{n+1} at first iteration $l = 0$, is given by

$$J^0_{n+1} = \Bigg[I - h\beta_0 \frac{\partial f(t_{n+1}, y^0_{n+1})}{\partial y} - h^2\gamma_0 \frac{\partial \frac{d}{dt} f(t_{n+1}, y^0_{n+1})}{\partial y} - h^3\delta_0 \frac{\partial \frac{d^2}{dt^2} f(t_{n+1}, y^0_{n+1})}{\partial y} - h^4\eta_0 \frac{\partial \frac{d^3}{dt^3} f(t_{n+1}, y^0_{n+1})}{\partial y} \Bigg]. \quad (38)$$

We do not need to update the J^0_{n+1} for the modified Newton–Raphson method to obtain good results.

As a first guess, we use

$$y^0_{n+1} = y^P_{n+1}, \quad (39)$$

obtained from predictor (28) [21]. At each iteration of the modified Newton–Raphson method, the derivatives $y^j_{n+1} = \frac{d^j}{dt^j} f\left(t_{n+1}, y_{n+1}\right)$ for $2 \leq j \leq 4$, of the Taylor series are calculated by known recurrence formulas (see, for example, [23, pp. 46–49], [5]). In the rest of this chapter, we compare numerically our new methods with BDF(5), SDMM(9) and TDMM(p), $p = 9, 11, 13$.

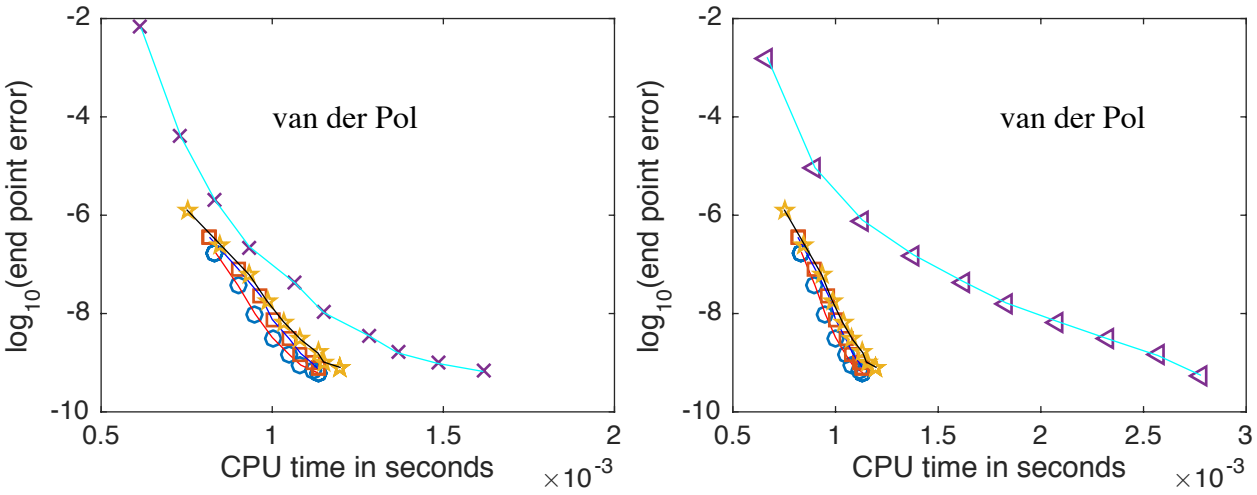
The necessary starting values at $t_1, t_2, \ldots, t_{k-1}$ for HBO$(4, p)$ were obtained by MATLAB's `ode15s` with stringent tolerance 5×10^{-14}.

4.4.2 Implementation and problems used for comparison

We compare the numerical performances of HBO$(4, p)$, BDF(5), SDMM(9) and TDMM(p) on each of the test problems listed below:

- **vdP,** the van der Pol oscillator (29).
- **OdB,** the Oregonator describing Belusov–Zhabotinskii reaction (40).
- **D1,** the Stiff DETEST problem D1(33).

In this chapter, we use the same five-step procedure as mentioned in (Chapter 3, Subection 3.4.2) for comparing the performance of the methods.



HBO(4,13) ∘, HBO(4,11) □, HBO(4,9) ⋆, SDMM(9) × and BDF(5) ◁

Figure 11: HBO$(4,p)$, $p = 9, 11, 13$, are compared with SDMM(9) (left) and BDF(5) (right) for Problem (vdP).

4.4.3 Comparing CPU time on the van der Pol oscillator

As a first comparison of HBO$(4,p)$ with BDF(5), SDMM(9) and TDMM(p), we consider the van der Pol oscillator equation used in [25] (see Subsection 3.4.3).

In Figure 11, we draw $\log_{10}$(EPE) as a function of CPU time for HBO$(4,p)$, $p = 9, 11, 13$, BDF(5) and SDMM(9). From this figure, we deduce that HBO$(4,p)$, $p = 9, 11, 13$, compare favorably with BDF(5) and SDMM(9) on Problem (vdP) at stringent tolerances.

Table 14 lists the CPU PEG and NST PEG, defined by Formulas (30) and (31), of HBO$(4,p)$, $p = 9, 11, 13$, over BDF(5) and SDMM(9) for problem (vdP). From the results, we can conclude that for $p = 9, 11$ and 13 HBO$(4,p)$ generally require less CPU time than BDF(5) and SDMM(9).

Comparing CPU time of HBO$(4,p)$ and TDMM(p) on the van der Pol oscillator

In Figure 12, we plot $\log_{10}$(EPE) (vertical axis) as a function of CPU time for HBO$(4,p)$ and TDMM(p), for $p = 9, 11, 13$. From this figure, we derive that HBO$(4,p)$ compare favorably with TDMM(p), for $p = 9, 11, 13$ on Problem (vdP) at stringent

Table 14: CPU PEG and NST PEG of the listed HBO$(4, p)$ over SDMM(9) and BDF(5) for Problem (vdP).

	CPU PEG of HBO$(4, p)$ over:		NST PEG of HBO$(4, p)$ over:	
HBO$(4, p)$	SDMM(9)	BDF(5)	SDMM(9)	BDF(5)
HBO(4,9)	20%	71%	146%	1 221%
HBO(4,11)	20%	63%	185%	1 430%
HBO(4,13)	20%	70%	215%	1 590%

Table 15: CPU PEG and NST PEG of the listed HBO$(4, p)$ over TDMM(p), for Problem (vdP).

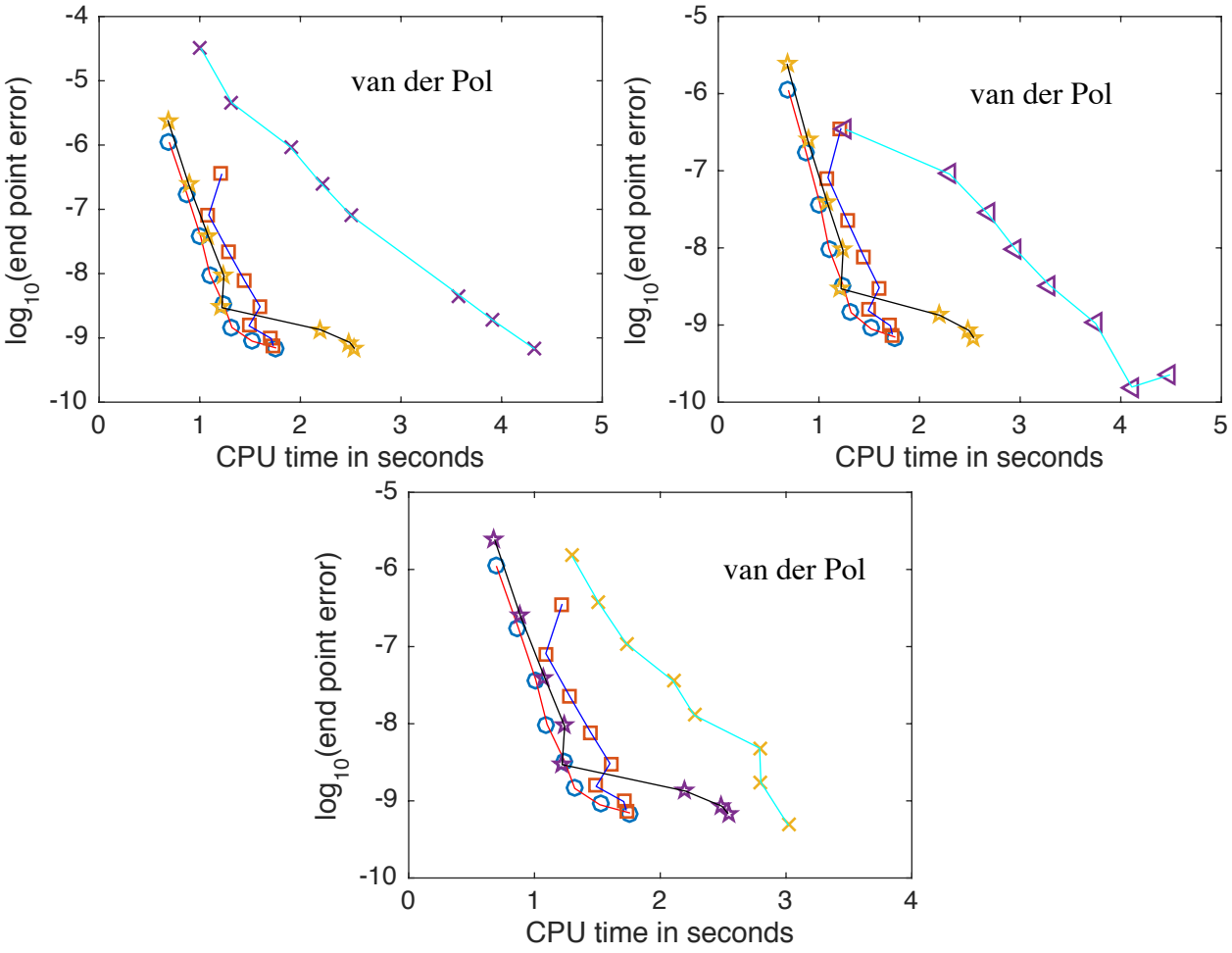
	CPU PEG of listed HBO over:			NST PEG of listed HBO over:		
HBO$(4, p)$	TDMM(9)	TDMM(11)	TDMM(13)	TDMM(9)	TDMM(11)	TDMM(13)
HBO(4,9)	116%	81%	53%	75%	37%	26%
HBO(4,11)	156%	117%	78%	113%	68%	50%
HBO(4,13)	185%	146%	101%	136%	86%	69%

tolerances.

Table 15 lists the CPU PEG and NST PEG, defined by Formulas (30) and (31), of HBO$(4, p)$, over TDMM(p), for $p = 9, 11, 13$ for problem (vdP). We can conclude that HBO$(4, p)$ generally need less CPU time than TDMM(p), for $p = 9, 11, 13$.

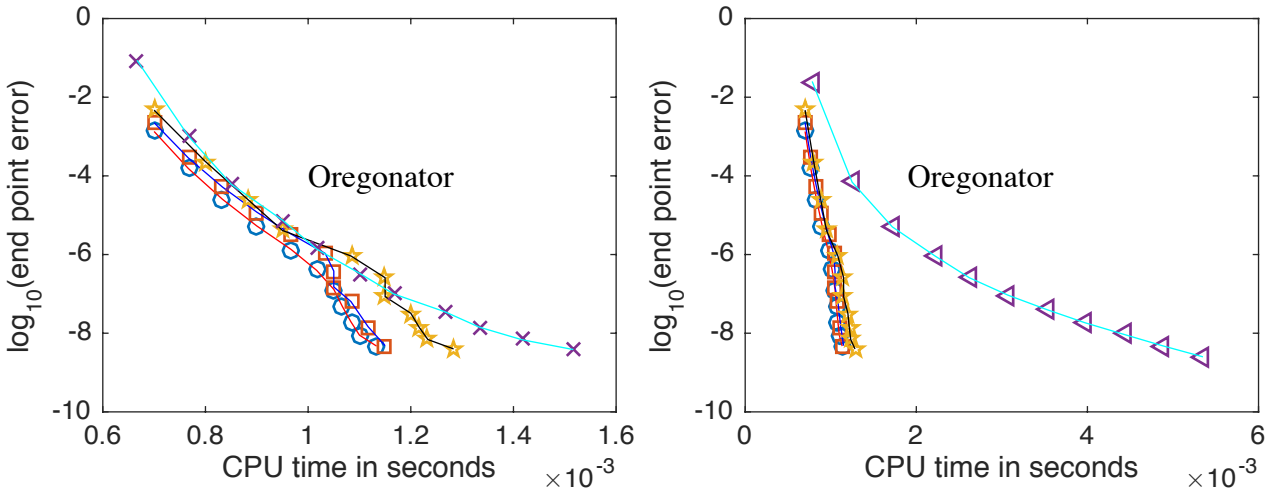
4.4.4 Comparing CPU time on the Oregonator describing Belusov–Zhabotinskii reaction

In 1984, Field and Noyes [19] have generalized a chemical reaction called Oregonator by a model . This model is one of the famous model equations in chemical reactions [27]. As a second comparison of HBO$(4, p)$ with BDF(5), SDMM(9) and TDMM(p), we consider the Oregonator describing the Belusov–Zhabotinskii reaction [19].



HBO(4,13) ∘, HBO(4,11) □, HBO(4,9) ⋆
TDMM(9) ×, TDMM(11) ◁ and TDMM(13) ×

Figure 12: HBO$(4, p)$, $p = 9, 11, 13$, are compared with TDMM(9) (top left), TDMM(11) (top right) and TDMM(13) (bottom) for Problem (vdP).



HBO(4,13) ∘, HBO(4,11) □, HBO(4,9) ⋆, SDMM(9) × and BDF(5) ◁

Figure 13: HBO$(4,p)$, $p = 9, 11, 13$, are compared with SDMM(9) (left) and BDF(5) (right) for Problem (OdB)

Problem 4 (OdB) *The Oregonator describing Belusov-Zhabotinskii reaction:*

$$\begin{aligned} y_1' &= 77.27(y_2 + y_1 - 8.375 \cdot 10^{-6} y_1^2 - y_1 y_2), & y_1(0) &= 1, \\ y_2' &= (y_3 - (1 + y_1)y_2)/77.27, & y_2(0) &= 2, \\ y_3' &= 0.161(y_1 - y_3), & y_3(0) &= 3, \end{aligned} \tag{40}$$

with $t_{end} = 20$.

In Figure 13, we plot $\log_{10}$(EPE) as a function of CPU time for HBO$(4,p)$, $p = 9, 11, 13$, BDF(5) and SDMM(9). From this figure, we deduce that for $p = 9, 11, 13$, HBO$(4,p)$ compare favorably with BDF(5) and SDMM(9) on problem (OdB) at stringent tolerances.

Table 16 lists for $p = 9, 11$ and 13 the CPU PEG and NST PEG, defined by Formulas (30) and (31), of HBO$(4,p)$ over BDF(5) and SDMM(9) for problem (OdB). From the results, we can conclude that for $p = 9, 11, 13$, HBO$(4,p)$ generally require less CPU time than BDF(5) and SDMM(9).

Comparing CPU time of HBO$(4,p)$ and TDMM(p) on the Oregonator reaction

In Figure 14, we draw for $p = 9, 11, 13$, $\log_{10}$(EPE) as a function of the CPU time

Table 16: CPU PEG and NST PEG of listed HBO$(4,p)$, $p = 9, 11, 13$, methods over SDMM(9) and BDF(5) for Problem (OdB).

	CPU PEG of listed HBO over:		NST PEG of listed HBO over:	
HBO$(4,p)$	SDMM(9)	BDF(5)	SDMM(9)	BDF(5)
HBO(4,9)	5%	132%	151%	1 552%
HBO(4,11)	10%	143%	200%	1 870%
HBO(4,13)	15%	148%	260%	2 125%

Table 17: CPU PEG and NST PEG of the listed HBO$(4,p)$ over TDMM(p) for Problem (OdB).

	CPU PEG of listed HBO over:			NST PEG of listed HBO over:		
HBO$(4,p)$	TDMM(9)	TDMM(11)	TDMM(13)	TDMM(9)	TDMM(11)	TDMM(13)
HBO(4,9)	67%	31%	5%	91%	58%	28%
HBO(4,11)	103%	58%	27%	132%	92%	55%
HBO(4,13)	138%	86%	49%	162%	117%	75%

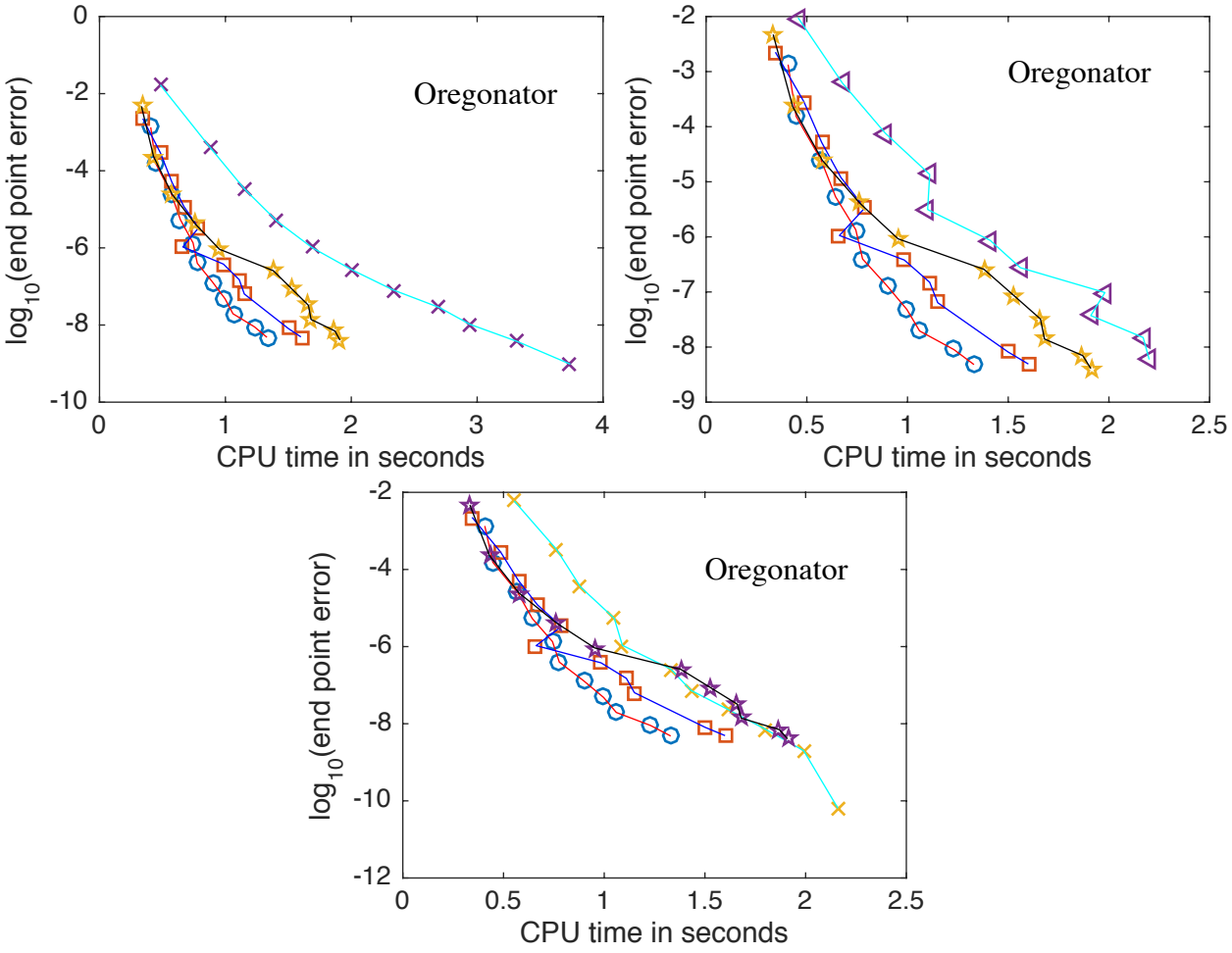
for HBO$(4,p)$ and TDMM(p). From this figure, we deduce that for $p = 9, 11, 13$, HBO$(4,p)$ compare favorably with TDMM(p) on Problem (OdB) at stringent tolerances.

Table 17 lists for $p = 9, 11$ and 13 the CPU PEG and NST PEG of HBO$(4,p)$ over TDMM(p) for problem (OdB). From the results, we can conclude that HBO$(4,p)$ often requires less CPU time than TDMM(p), for $p = 9, 11, 13$.

4.4.5 Comparing CPU time on a stiff DETEST problem

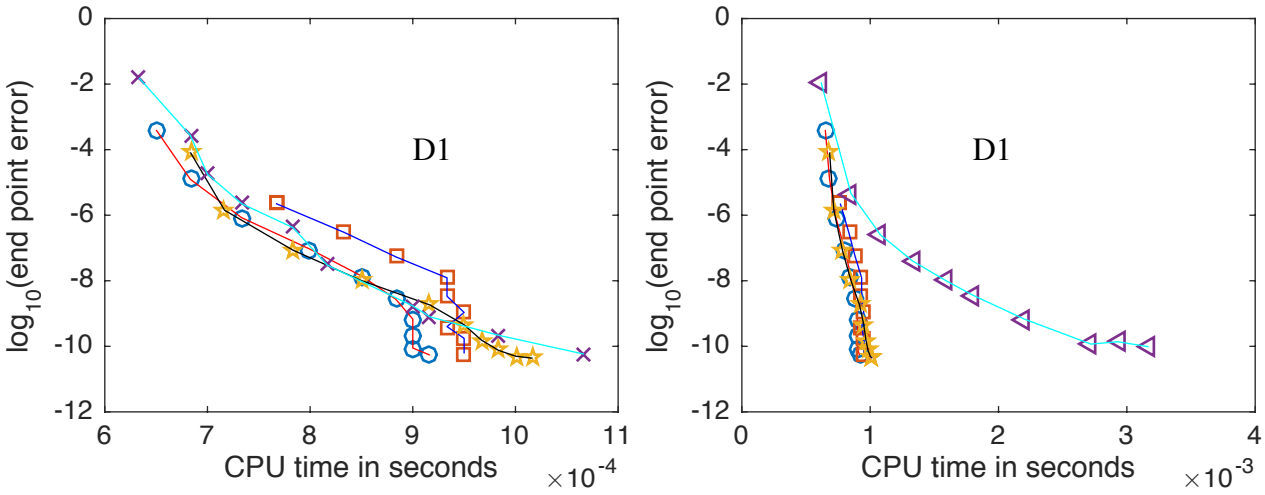
As a third comparison, we consider the stiff DETEST problem D1 [17] (see Subsection 3.4.5).

In Figure 15, we plot for $p = 9, 11$ and 13 $\log_{10}$(EPE) as a function of the CPU time for HBO$(4,p)$ BDF(5) and SDMM(9). From this figure we deduce that HBO$(4,p)$, $p = 9, 11, 13$, compare favorably with BDF(5) and SDMM(9) for Problem (D1) at stringent tolerances.



HBO(4,13) ∘, HBO(4,11) □, HBO(4,9) ⋆
TDMM(9) ×, TDMM(11) ◁ and TDMM(13) ×

Figure 14: HBO$(4, p)$, $p = 9, 11, 13$, are compared with TDMM(9) (top left), TDMM(11) (top right) and TDMM(13) (bottom) for Problem (OdB).



HBO(4,13) ∘, HBO(4,11) □, HBO(4,9) ⋆, SDMM(9) × and BDF(5) ◁

Figure 15: HBO$(4,p)$, $p = 9, 11, 13$, are compared with SDMM(9) (left) and BDF(5) (right) for Problem (D1).

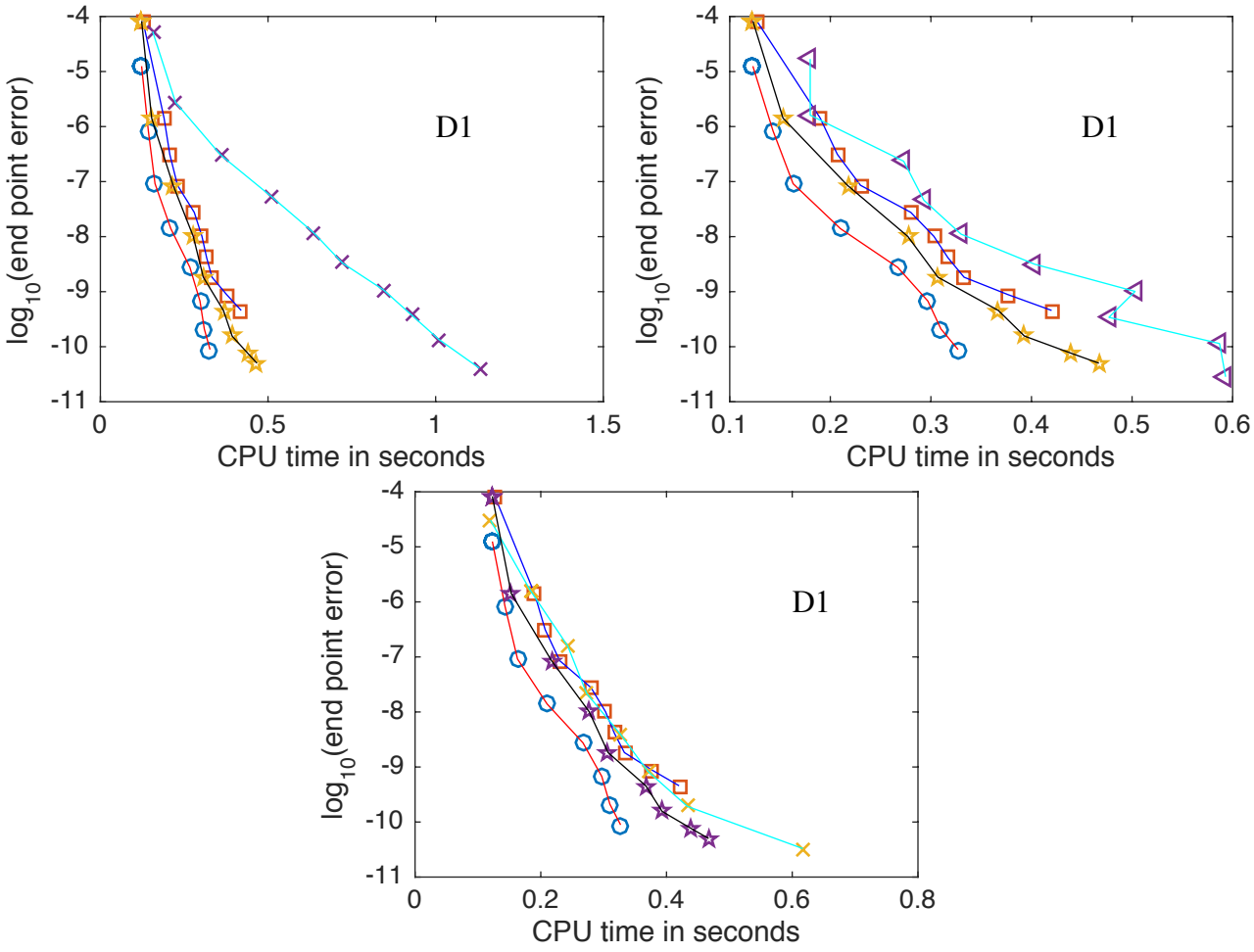
Table 18 lists for $p = 9, 11$ and 13 the CPU PEG and NST PEG of HBO$(4,p)$ over BDF(5) and SDMM(9) for problem (D1). We can conclude that, in most cases HBO$(4,p)$, $p = 9, 11, 13$, requires less CPU time than BDF(5) and SDMM(9) .

Table 18: CPU PEG and NST PEG of HBO$(4,p)$, $p = 9, 11, 13$, over BDF(5) and SDMM(9) for Problem (D1).

	CPU PEG of listed HBO over:		NST PEG of listed HBO over:	
HBO$(4,p)$	SDMM(9)	BDF(5)	SDMM(9)	BDF(5)
HBO(4,9)	3%	80%	163%	2 279%
HBO(4,11)	-3%	63%	238%	2 956%
HBO(4,13)	5%	83%	294%	3 455%

Comparing CPU time of HBO$(4,p)$, TDMM(p) on the stiff DETEST problem D1

In Figure 16, we draw for $p = 9, 11, 13$ $\log_{10}$(EPE) as function of CPU time for HBO$(4,p)$ and TDMM(p). From this figure, we derive that for $p = 9, 11, 13$, HBO$(4,p)$ compare favorably with TDMM(p) for Problem (D1) at stringent tolerances.



HBO(4,13) ∘, HBO(4,11) □, HBO(4,9) ⋆
TDMM(9) ×, TDMM(11) ◁ and TDMM(13) ×

Figure 16: HBO$(4, p)$, $p = 9, 11, 13$, are compared with TDMM(9) (top left), TDMM(11) (top right) and TDMM(13) (bottom) for Problem (D1).

Table 19: CPU PEG and NST PEG of the listed HBO$(4, p)$ over TDMM(p) for Problem (D1).

	CPU PEG of listed HBO over:			NST PEG of listed HBO over:		
HBO$(4, p)$	TDMM(9)	TDMM(11)	TDMM(13)	TDMM(9)	TDMM(11)	TDMM(13)
HBO(4,9)	141%	32%	15%	95%	58%	19%
HBO(4,11)	100%	19%	3%	99%	52%	27%
HBO(4,13)	199%	63%	43%	205%	126%	86%

Table 19 lists for $p = 9, 11$ and 13 the CPU PEG and NST PEG of HBO$(4, p)$ over TDMM(p) for problem (D1). From the results, we can conclude that HBO$(4, p)$ require less CPU time than TDMM(p), for $p = 9, 11, 13$.

Chapter 5

Conclusion

In this thesis, we construct two new families of Hermite–Birkhoff–Obrechkoff methods: they are, respectively, implicit three–and four-derivative methods of order $5 \leq p \leq 14$ and $7 \leq q \leq 14$, denoted by HBO$(3, p)$ and HBO$(4, q)$.
These methods are designed to solve stiff systems of first-order differential equations. We determine their stability properties and test their efficiency.

The methods HBO$(3, p)$ and HBO$(4, q)$ are $A(\alpha)$-stable, their regions of stability are comparable to those of known methods (described in Chapter 2). For HBO$(3, p)$, (respectively HBO$(4, q)$) the angle α is, in general, slightly smaller (larger respectively) than that of comparable methods. Remark that, in general, the higher the order of derivatives is, the larger the stability region becomes. For $p = 5, 6$ and $q = 7, 8$, HBO$(3, p)$ and HBO$(4, q)$ are A-stable. In a future work, we propose to study and show that these A-stable methods will be suitable for the integration of stiff differential systems whose Jacobians have some large eigenvalues lying close to the imaginary axis.
We determine the efficiency of our methods by comparing their CPU PEG (the CPU percentage efficiency gain) and NST PEG (the number of steps percentage efficiency gain), for a family of test problems.
More precisely, the performance of HBO$(3, p)$ and well-known methods is compared on

- Van der Pol oscillator,
- Robertson chemical reaction,
- Stiff DETEST problem D1,

similarly, the performance of HBO$(4, p)$ and well-known methods is compared on

- Van der Pol oscillator,
- Oregonator describing Belusov–Zhabotinskii reaction,
- Stiff DETEST problem D1.

Our results, described in Chapter 3 and 4, show that

- HBO$(3, p)$ and HBO$(4, p)$ are more efficient than several other well known methods.
- Third and fourth derivative methods are generally more efficient than first and second derivative methods. However, the task of computing a Jacobian of stiff differential systems involving third and fourth derivatives might be, unfortunately, very laborious (even nasty).

Chapter 6

Coefficients for HBO$(3, p)$ and HBO$(4, p)$

6.1 Coefficients of new 3-derivative HBO$(3, p)$ and 4-derivative HBO$(4, p)$

6.1.1 Coefficients of HBO$(3, p)$, of order $p = 5, 6, \ldots, 13$.

Table 20: Coefficients of HBO$(3, p)$ of order $p = k + 4 = 5, 6, 7$.

k	1	2	3
coeffs$\backslash p$	5	6	7
δ_0	1.6666666666666607e-02	1.2499999999999734e-02	1.0515873015872895e-02
γ_0	-1.4999999999999966e-01	-1.2916666666666540e-01	-1.1792328042327985e-01
γ_1	5.0000000000000239e-02	6.6666666666667623e-02	7.8571428571428875e-02
β_0	5.9999999999999942e-01	5.6458333333333122e-01	5.4397045855379100e-01
β_1	4.0000000000000058e-01	4.3333333333333529e-01	4.5119047619047720e-01
β_2		2.0833333333334226e-03	5.0595238095237551e-03
β_3			-2.2045855379187725e-04

Table 21: Coefficients of HBO$(3, p)$ of order $p = 8, 9, 10$.

k	4	5	6
coeffs$\backslash p$	8	9	10
δ_0	9.3005952380951773e-03	8.4589947089946382e-03	7.8317901234568332e-03
γ_0	-1.1042906746031719e-01	-1.0490255731922390e-01	-1.0057484567901248e-01
γ_1	8.8293650793650674e-02	9.6709656084655621e-02	1.0423611111111047e-01
β_0	5.2947151951058169e-01	5.1832545561434462e-01	5.0929754249951054e-01
β_1	4.6253306878306916e-01	4.7024774029982380e-01	4.7564169973544962e-01
β_2	8.7053571428569020e-03	1.2913359788359412e-02	1.7617394179893716e-02
β_3	-7.6058201058197409e-04	-1.6956937095825317e-03	-3.0894816774445504e-03
β_4	5.0636574074071437e-05	2.2597001763667530e-04	6.1797288359786691e-04
β_5		-1.6832010582009943e-05	-9.2096560846558655e-05
β_6			6.9689398393100773e-06

6.1.2 Coefficients of HBO$(4, p)$, of order $p = 5, 6, \ldots, 13$.

Table 22: Coefficients of HBO$(3, p)$ of order $p = 11, 12, 13$.

k	7	8	9
coeffs$\backslash p$	11	12	13
δ_0	7.3409191117524142e-03	6.9430634469692387e-03	6.6120102386191537e-03
γ_0	-9.7047586837765515e-02	-9.4089245787484899e-02	-9.1554061337191572e-02
γ_1	1.1110830527497154e-01	1.1747399591150204e-01	1.2343295366179957e-01
β_0	5.0172810557808356e-01	4.9522321169493949e-01	4.8952884305001004e-01
β_1	4.7942140652557302e-01	4.8201315199901801e-01	4.8369442936428481e-01
β_2	2.2771539802789637e-02	2.8341519109753490e-02	3.4300476860046981e-02
β_3	-4.9984245007393454e-03	-7.4739708593898014e-03	-1.0563800803984648e-02
β_4	1.3338264423334136e-03	2.4942387979507512e-03	4.2322681417844739e-03
β_5	-2.9826238576239581e-04	-7.4386073031943215e-04	-1.5781148153592522e-03
β_6	4.5147796305205850e-05	1.6892511423771068e-04	4.7790810869678528e-04
β_7	-3.3392585830229488e-06	-2.4991267550807174e-05	-1.0606552265665973e-04
β_8		1.7761413606384832e-06	1.5077386338938537e-05
β_9			-1.0217691615701147e-06

Table 23: Coefficients of HBO$(4, p)$ of order $p = k + 6 = 7, 8, 9$.

k	1	2	3
coeffs$\backslash p$	7	8	9
η_0	-1.1904761904759364e-03	-8.9285714285746581e-04	-7.4955908289253692e-04
δ_0	1.9047619047616766e-02	1.5922619047622849e-02	1.4274691358025941e-02
δ_1	4.7619047619054319e-03	6.5476190476162766e-03	7.8373015873009003e-03
γ_0	-1.4285714285713391e-01	-1.2901785714287511e-01	-1.2125587889477346e-01
γ_1	7.1428571428577156e-02	8.2142857142842071e-02	8.8591269841265197e-02
β_0	5.7142857142855674e-01	5.4665178571431916e-01	5.3213489613953602e-01
β_1	4.2857142857144331e-01	4.5357142857139482e-01	4.6840277777776779e-01
β_2		-2.2321428571386254e-04	-5.4563492063500923e-04
β_3			7.9610033313860251e-06

Table 24: Coefficients of HBO$(4, p)$ of order $p = 10, 11, 12$.

k	4	5	6
coeffs$\backslash p$	10	11	12
η_0	-6.6137566137560583e-04	-6.0014830848156842e-04	-5.5444491208368057e-04
δ_0	1.3194444444444058e-02	1.2407672959755429e-02	1.1797532617844213e-02
δ_1	8.8955026455034211e-03	9.8139129389142668e-03	1.0636574074075401e-02
γ_0	-1.1593915343915141e-01	-1.1193294569370511e-01	-1.0873911618109663e-01
γ_1	9.3176807760142935e-02	9.6697380551551096e-02	9.9521850448937771e-02
β_0	5.2187224426807322e-01	5.1394545693096283e-01	5.0749617068431974e-01
β_1	4.7904357730747121e-01	4.8739855983780600e-01	4.9431759701847522e-01
β_2	-9.4246031746095119e-04	-1.4016654641663567e-03	-1.9158286736420612e-03
β_3	2.7557319224024748e-05	6.1572515276276819e-05	1.1235406682943317e-04
β_4	-9.1857730746743866e-07	-4.1075019373660849e-06	-1.1248657624528746e-05
β_5		1.8368205868216327e-07	1.0063431938433069e-06
β_6			-5.0781551553157219e-08

Table 25: Coefficients of HBO$(4, p)$ of order $p = 13$.

k	7
coeffs$\backslash p$	13
η_0	-5.1863792439185014e-04
δ_0	1.1304163480289571e-02
δ_1	1.1388520815605201e-02
γ_0	-1.0609590159214446e-01
γ_1	1.0185288534768008e-01
β_0	5.0206578190475215e-01
β_1	5.0025338104647343e-01
β_2	-2.4797887297894157e-03
β_3	1.8197876511923071e-04
β_4	-2.4303288553865888e-05
β_5	3.2621834184327797e-06
β_6	-3.2928034471235219e-07
β_7	1.7398925020357870e-08

Bibliography

[1] R. Ashino, M. Nagase and R. Vaillancourt, Behind and Beyond the MATLAB Suite, an international journal computer and mathematics with application. 40 (2000), 491–512.

[2] B. K. Aliyu, C. A. Osheku, A. A. Funmilayo and J. I. Musa, Identifying Stiff Ordinary Differential Equations and Problem Solving Environments (PSEs), Sciencedomain international. 11 (2014), 1430–1448.

[3] A. Björck and V. Pereyra, Solution of Vandermonde systems of equations, Math. Comp. 24 (1970), 893–903.

[4] A. Björck and T. Elfving, Algorithms for confluent Vandermonde systems, Numer. Math. 21 (1973), 130–137.

[5] R. Barrio, F. Blesa and M. Lara, VSVO formulation of the Taylor method for the numerical solution of ODEs, Comput. Math. Appl. 50 (2005), 93–111.

[6] C. Baker, G. Monegato, J. Pryce and G. V. Berghe, Numerical Analysis 2000, Volume 6 Ordinary Differential Equations and Integral Equations, Elsevier Science B.V. North-Holland, 2001.

[7] A. Buscarino, L. Fortuna, M. Frasca and G. Sciuto, A Concise Guide to Chaotic Electronic Circuits, Springer Science and Business, 2014.

[8] C. F. Curtiss and J. O. Hirschfelder, Integration of Stiff Equations, Proc Natl. Acad. Sci. USA, (1952), 235–243.

[9] C. W. Cryer, On the instability of high order backward-difference multistep methods, BIT, 12 (1972), 17–25.

[10] J. R. Cash, On the integration of stiff systems of ODEs using extended backward differentiation formula, Numer. Math. 34 (1980), 235–246.

[11] J. R. Cash, Second derivative extended backward differentiation formulas for the numerical integration of stiff systems, SIAM J. Numer. Anal. 18 (1981), 21–36.

[12] A. Deprit and R. M. W. Zahar, Numerical integration of an orbit and its concomitant variations, Z. Angew. Math. Phys. 17 (1966), pp. 425430.

[13] A. K. Datta, An evaluation of the approximate inverse algorithm for numerical integration of stiff differential equations, Imperial Chemical Industries Ltd. Cheshire. (1967).

[14] L. Edsberg, Integration Package for Chemical Kinetics, Plenum Press. New York. (1974), 81–94.

[15] W. H. Enright, Second derivative multistep methods for stiff ordinary differential equations, SIAM J. Numer. Anal. 11 (1974), 321–331.

[16] W. H. Enright, T. E. Hull and B. Lindberg Comparing Numerical Methods for stiff systems of O.D.E:s, BIT Numerical Mathematics. (1975), 10–48.

[17] W. H. Enright and J. D. Pryce, Two Fortran packages for assessing initial value methods, ACM Trans. on Math. Software 13 (1987), 1–27.

[18] A. K. Ezzeddine and G. Hojjati, Third Derivative Multistep Methods for Stiff Systems, Int. J. Nonlinear Sci. 14 (2012), 443–450.

[19] J. Field and R. M. Noyes, Oscillations in chemical systems. IV: Limit cycle behavior in a model of a real chemical reaction, J. Chem. Phys. 60 (1974), 1877–1884.

[20] G. Galimberti and V. Pereyra, Solving confluent Vandermonde systems of Hermite type, Numer. Math. 18, (1971), 44–60.

[21] C. W. Gear, Algorithm 407, DIFSUB for solution of ordinary differential equations, Comm. ACM. 14 (1971), 185–190.

[22] C. W. Gear, Numerical Initial Value Problems in Ordinary Differential Equations, Prentice-Hall, Englewood Cliffs, N.J, 1971.

[23] E. Hairer, S. P. Nørsett and G. Wanner, Solving Ordinary Differential Equations I. Nonstiff Problems, corr. sec. print., Section III.8, Springer-Verlag, Berlin, 2000.

[24] E. Hairer and G. Wanner, Solving Ordinary Differential Equations II. Stiff and Differential-Algebraic Problems, corr. sec. print., Springer-Verlag, Berlin, 2002.

[25] G. Hojjati, M. Y. Rahimi and S. M. Hosseini, New second derivative multistep methods for stiff systems, Appl. Math. Model. 30 (2006), 466–476.

[26] G. Ismail and I. Ibrahim, New efficient second derivative multistep methods for stiff systems, Appl. Math. Model. 23 (1999), 279–288.

[27] M. Idrees, F. Mabood, A. Ali and G. Zaman, Exact Solution for a Class of Stiff Systems by Differential Transform Method, Applied Mathematics. 4 (2013), 440–444.

[28] R. W. Klopfenstein, Numerical differentiation formulas for stiff systems of ordinary differential equations, RCA reviews. 32 (1971), 447–462.

[29] P. Kaps and A. Ostermann, Rosenbrock Methods using few LU-Decompositions, IMA J Numer Anal. 9 (1) (1989), 15–27.

[30] M. M. Khalsaraei, N. N. Oskuyi and G. Hojjati, A class of second derivative multistep methods for stiff systems, Acta Universitatis Apulensis. (2012) 171–188.

[31] S. Kinoshita, Pattern Formations and Oscillatory Phenomena , Elsevier, 2013.

[32] J. D. Lambert, Numerical Methods for Ordinary Differential Systems, Wiley, Chichester UK, 1991.

[33] W. Liniger and R. Willoughby, Efficient numerical integration of stiff systems of ordinary differential equations, Tech. Report RC-1970. IBM Thomas J. Watson Research Center. Yorktown Heights, New York. 1967.

[34] N. Obrechkoff, Neue Quadraturformeln, Abh. Preuss. Akad. Wiss. Math. Nat. Kl. 4 (1940), 1–20.

[35] N. Obrechkoff, Sur les quadrature mecaniques, Spisanic Bulgar. Akad. Nauk. 65 (1942), 191– 289.

[36] L. Petzold, Automatic selection of methods for solving stiff and nonstiff systems of ordinary differential equations, SIAM J. Sci. and Stat. Comput. 4(1) (1983), 136–148.

[37] G. Psihoyios, A Class of Implicit Advanced Step-Point Methods with a Parallel Feature for the Solution of Stiff Initial Value Problems, Elsevier Ltd. (2004), 1199–1224.

[38] E. Rabe, Determination and survey of periodic Trojan orbits in the restricted problem of three bodies, Astronomical J. 66 (1961), 500–513.

[39] H. H. Robertson, The solution of a set of reaction rate equations, In: J. Walsh ed: Numer. Anal. an Introduction, Academic Press (1966), 178–182.

[40] T. Reiher, Statilitatsuntersuchungen bei rückwärtigen Differentiationsformeln in Abhängigkeit von einem Parameter, Tech. Report 11. Sektion Mathematik. Humboldt–Universität zu Berlin, 1978.

[41] J. F. Steffensen, On the restricted problem of three bodies, Danske Vid. Selsk., Mat.-fys. Medd. 30(18) (1956), 1–17.

[42] P. W. Sharp, Numerical comparison of explicit Runge–Kutta pairs of orders four through eight, ACM Trans. on Math. Software 17 (1991), 387–409.

[43] A. M. Stuart and A. R. Humphries, Dynamical Systems and Numerical Analysis, Cambridge University Press, New York, 1996.

[44] L. F. Shampine and M. W. Reichelt, The MATLAB ODE suite, SIAM J. Sci. Comput. 18(1) (1997), 1–22.

[45] E. Suli and D. F. Mayers, An Introduction to Numerical Analysis, Cambridge University Press, 2003.

[46] V. Satek, Stiff Systems Analysis, Information Sciences and Technologies Bulletin of the ACM Slovakia. Vol. 4, No. 3 (2012), 1-11.

[47] A. M. Winslow, Extrapolant formulation of the backward differentiation method with application to chemical kinetic equations, J. Phys. Chem. 81 (25) (1977), 2409–2413.

[48] Wikipedia, Stiff equation, 28 October 2015.