Department of Mathematics
The University of Auckland
New Zealand

Efficient Numerical Integration for Gravitational N-Body Simulations

Muhammad Amer Qureshi

January 2012

Supervisor: Philip W. Sharp

A thesis submitted in fulfillment of the requirements of Doctor of Philosophy in Applied Mathematics

Abstract

Models for N-body gravitational simulations of the Solar System vary from small simulations of two bodies over short intervals of time to simulations of large numbers of bodies over long-term integration. Most simulations require the numerical solution of an initial value problem (IVP) of second-order ordinary differential equation. We present new integration methods intended for accurate simulations that are more efficient than existing methods.

In the first part of the thesis, we present new higher-order explicit Runge–Kutta Nyström pairs. These new pairs are searched using a simulated annealing algorithm based on optimisation. The new pairs are up to approximately 60% more efficient than the existing ones. We implement these new pairs for a variety of gravitational problems and investigate the growth of global error in position for these problems along with relative error in conserved quantities.

The second part consists of the implementation of the Gauss Implicit Runge–Kutta methods in an efficient way such that the error growth satisfies Brouwer's Law. Numerical experiments show that using the new way of implementation reduces the integration cost up to 20%. We also implement continuous extensions for the Gauss implicit Runge–Kutta methods, using interpolation polynomials at nodal points.

Acknowledgements

First and above all, I praise Omnipresent Allah, the Almighty, for providing me with this opportunity and granting me the capability and strength to strive successfully in seeking knowledge.

This thesis appears in its current form due to the assistance and guidance of several people. I would therefore like to offer my sincere thanks to all of them.

To Dr. Philip W. Sharp, my esteemed supervisor, my cordial thanks for accepting me as a Ph.D student, and for his warm encouragement, thoughtful guidance, critical comments, correction of the thesis and continual assistance throughout the period of study.

I would also like to thank all of my friends studying at the University of Auckland especially my colleagues Dr. Yousaf Habib, Shafiq ur Rehman, Saghir Ahmed, Attique ur Rehman, Gulshad Imran and particularly Bashir Hussain for their support and useful discussions.

In terms of financial assistance, I have been very fortunate to have received the scholarship funded by the Higher Education Commission of Pakistan (HEC). I also wish to thank the University of Auckland for funding me to attend local and international conferences.

I am also immensely grateful to my brothers Masood and Kashif for their continual support, understanding and encouragement.

Lastly, I do not have words to thank and describe that what they have done for me, the people who deserve the most acknowledgement and to whom I dedicate this dissertation: my parents. I am greatly indebted to my parents for prayers which are the essential ingredients towards the completion of this thesis.

Muhammad Amer Qureshi
The University of Auckland
New Zealand.

Contents

List of Figures

List of Tables

1

Introduction

1.1 Introduction

Many physical phenomena in science and engineering are described through the process of mathematical modelling. These models include those in climatology, mathematical biology, computational finance and dynamical astronomy. The models are often expressed in terms of the unknown quantities and their derivatives in the form of ordinary and partial differential equations. In this thesis, we are concerned with the solution of ordinary differential equations (ODEs) that arise when doing N-body gravitational simulations of the Solar System. The solution of these differential equations give valuable insight into the evolution of the Solar System. In some cases, these equations can be solved analytically, but most of the differential equations are too complicated to possess analytical solutions. This necessitates the use of approximation techniques to find the numerical solutions. The main goal of this thesis is to present new efficient methods for doing accurate simulations.

N-body gravitational simulations of the Solar System vary from small simulations of two bodies over short intervals of time to simulations of large numbers of bodies over long intervals. Most simulations require the numerical solution of an initial value problem (IVP) of second order ordinary differential equation. The IVP often takes the form

$$y''(x) = f(x, y(x)), \qquad y(x_0) = y_0, \qquad y'(x_0) = y'_0, \tag{1.1.1}$$

where $'$ denotes the differentiation with respect to time x and $f : \mathbb{R} \times \mathbb{R}^n \longrightarrow \mathbb{R}^n$, where n is the dimension of the problem. These IVPs are usually in the autonomous form

$$y''(x) = f(y(x)), \qquad y(x_0) = y_0, \qquad y'(x_0) = y'_0, \tag{1.1.2}$$

where $f : \mathbb{R}^n \longrightarrow \mathbb{R}^n$.

Physical systems often have conserved quantities. These include the total energy H, the angular momentum L, and position and velocity of the center of mass of the bodies. The quantities will usually not be conserved exactly by the numerical solution and the error provides insight about the accuracy of the solution.

1.2 Kepler's two-body problem

One of the simplest models is Kepler's two-body problem. It is a commonly used problem for planetary orbital problems because it has an analytical solution. Kepler's problem defines the motion of one body orbiting another. The equations of motion can be written as

$$\begin{aligned} y''_1 &= -y_1/r^3, \\ y''_2 &= -y_2/r^3, \end{aligned} \tag{1.2.1}$$

where y_1 and y_2 are the coordinates of one body relative to the other, $r = \sqrt{y_1^2 + y_2^2}$, and the initial conditions are $y_1(0) = 1-e$, $y'_1(0) = 0$, $y_2(0) = 0$ and $y'_2(0) = (1+e)^{1/2}(1-e)^{-1/2}$. The parameter e is the orbital eccentricity ($0 \leq e < 1$). The exact solution of the above equations (1.2.1) is

$$y_1 = \cos(E) - e, \qquad y_2 = \sqrt{1 - e^2}\sin(E),$$

and

$$y_1' = -\sin(E)(1-e\cos(E))^{-1}, \qquad y_2' = \sqrt{(1-e^2)}\cos(E)(1-e\cos(E))^{-1},$$

where the eccentric anomaly E satisfies Kepler's equation $x = E - e\sin(E)$. Since Kepler's equation is implicit in E, the equation is usually solved using a non-linear equation solver, although useful analytical approximations can be found for smaller eccentricity.

The total energy, also called the Hamiltonian H, and the angular momentum L for Kepler's problem are

$$H = \frac{y_1'^2 + y_2'^2}{2} - \frac{1}{\sqrt{y_1^2 + y_2^2}}, \tag{1.2.2}$$

$$L = y_1 y_2' - y_2 y_1'. \tag{1.2.3}$$

Methods that attempt to conserve the Hamiltonian and other properties are discussed later in Chapter 2.

More realistic models are obtained by increasing the number of bodies. This increased realism comes at the cost of usually having to find the solution numerically. We now describe four such problems. These will be used later in the thesis as test problems.

1.3 Jovian Problem

The Jovian Problem models the orbital motion of the Sun and the four gas giants Jupiter, Saturn, Uranus and Neptune when these bodies are treated as point masses. Let $\mathbf{r}_i$, $i = 1, \ldots, 5$, denote the position of the i^{th} body in three-dimensional Cartesian coordinates with the origin at the barycentre (centre of mass) of the bodies. The equations of motion of the bodies can be written as

$$\mathbf{r}_i''(x) = \sum_{j=1, j\neq i}^{5} \frac{\mu_j(\mathbf{r}_j(x) - \mathbf{r}_i(x))}{\|\mathbf{r}_j(x) - \mathbf{r}_i(x)\|_2^3}, \qquad i = 1, \ldots, 5, \tag{1.3.1}$$

where $\|.\|_2$ denotes the L_2 norm, $\mu_j = Gm_j$, G being the gravitational constant and m_j the mass of the j^{th} body. The value of the μ_j are given in Appendix A. The independent

variable x is in units of days. Throughout the thesis we will assume that the bodies are ordered Sun, Jupiter, Saturn, Uranus and Neptune.

The initial conditions we use are given in Appendix A and are taken from Sharp (private communication). The initial conditions were chosen so that the barycentre is at the origin and has zero velocity at $x = 0$. The analytical solution will then satisfy

$$\sum_{i=1}^{5} \mu_i r_i = 0, \qquad \sum_{i=1}^{5} \mu_i r_i' = 0.$$

The Hamiltonian H and angular momentum L for the analytical solution are given by the expressions

$$H = \frac{1}{2}\sum_{i=1}^{5} m_i \mathbf{r}_i' \cdot \mathbf{r}_i' - G \sum_{i=1, j\neq i}^{5} \sum_{j=1}^{5} \frac{m_i m_j}{\|\mathbf{r}_j(x) - \mathbf{r}_i(x)\|_2}, \tag{1.3.2}$$

$$L = \sum_{i=1}^{5} \mu_i (\mathbf{r}_i \times \mathbf{r}_i'). \tag{1.3.3}$$

Simulations of these bodies are of significant importance as they have a prime role in the dynamics of the Solar System. For example, Jupiter, by sweeping up debris which could have bombarded the Earth, was crucial to the evolution of life on Earth [80].

The numerical integrations of the Jovian Problem were first done by Cohen and Hubard [15], Kinoshita and Nikai [57], Applegate *et al.* [3] and Sussman and Wisdom [88]. Afterwards Grazier *et al.* [39], Sussman and Wisdom [89], Laskar [64] and Sharp [81] integrated for larger time scales. These long-term integrations provided insight into the Solar System dynamics that went further than that given by analytical theories.

1.4 Nine Planets Problem

The Nine Planets Problem is the Jovian Problem with the addition of the terrestrial planets Mercury, Venus, Earth, Mars and the dwarf planet Pluto. The equations of motion and the expressions for the conserved quantities are the same as for the Jovian Problems except the number of bodies is ten and not five. The initial conditions and the

Planet	Orbital period	Eccentricity	Semi-major axis (AU)	Mass
Mercury	1.00	0.206	0.3075	1.00
Venus	2.55	0.007	0.723	14.74
Earth	4.15	0.017	1.000	18.08
Mars	7.81	0.093	1.523	1.94
Jupiter	49.25	0.048	5.204	5723.8
Saturn	122.308	0.056	9.582	1720.1
Uranus	348.81	0.047	19.229	202.3
Neptune	684.21	0.009	30.103	308.9
Pluto	1029.32	0.248	39.481	0.039

Table 1.1: Some orbital and physical data for the eight planets and Pluto. The semi-major axis is in astronomical units (AU) and the mass in units of Mercury's mass.

values of μ for the extra five bodies are given in Appendix A.

Table 1.1 lists some orbital and physical data for the planets in the Nine Planets Problem. The orbital period and mass are expressed in units of Mercury's orbital period and mass, and the semi-major axis of the orbits are in astronomical units (a standardised value for the distance of Earth from the Sun). We observe from the table that the ratio of the longest to shortest orbital period is over 1000, and that the eccentricity varies from the near-circular value 0.007 for Venus to the eccentric value of 0.248 for Pluto. We also observe that the four gas giants are at least 10 times as massive as Earth.

Richardson and Walker [74], Quinn *et al.* [73] and Laskar [63, 64] among others have integrated the Nine Planets Problem. The shortest orbital period for the problem is 88 days for Mercury. This is approximately 50 times smaller than the shortest orbital period of the Jovian Problem. So the average step-size for the Nine Planets Problem should be about 50 times smaller than that used for integration of the Jovian Problem [81].

We use the Earth-Moon system in place of Earth i.e. including the mass of Moon with that of Earth and take the Earth-Moon barycenter as the position of Earth to make the problem more realistic and consistent with Sharp [81].

1.5 Helin-Roman-Crockett Problem

The Helin-Roman-Crockett (HRC) problem models a comet having multiple close approaches with Jupiter. The equations of motion are the same as those for the Jovian Problem with the addition of the following equations for the position $\mathbf{r}_6$ of the comet

$$\mathbf{r}_6''(x) = \sum_{j=1, j\neq i}^{5} \frac{\mu_j(\mathbf{r}_j(x) - \mathbf{r}_6(x))}{\|\mathbf{r}_j(x) - \mathbf{r}_6(x)\|_2^3}. \tag{1.5.1}$$

The initial conditions for the comet are given in Appendix A.

In this problem, the comet has five close approaches with the Jupiter over an interval of approximately 5600 days. Figures 1.1 and 1.2 are produced doing a simulation for an interval of 7000 days. Figure 1.1 is a phase-plane plot in the two dimensional $y_1 - y_2$ plane of the position of the comet relative to Jupiter for $x = 1000$ days to $x = 8000$. This plot is sometimes called the tulip diagram because of its similarities to the petals of a tulip. Figure 1.2 gives the graph of the distance of the comet from Jupiter for the same interval of x. This clearly shows the five close approaches. There is also a sixth time where the distance is a local minimum. The distance from Jupiter at this local minimum is significantly larger than that for the first five local minima. For this reason, the sixth local minimum is often not regarded as a close approach.

The close approach of the comet necessitates the use of smaller step-size at the time of close approach. Figure 1.3 contains the graph of the average step-size versus time for the explicit Runge–Kutta Nyström pairs of Dormand *et al.* [20, 21] having orders 4-6, 6-8 and 10-12. The interval of integration is 8000 days. We observe from the Figure 1.3 that the average step-size decreases significantly as the comet makes a close approach with the Jupiter. The red, green and blue graphs were obtained using the explicit Runge–Kutta Nyström 4-6, 6-8 and 10-12 pairs respectively. The integrations were performed in double precision using the severe local error tolerances of 10^{-14}.[1]

[1] We are not advocating that the 4-6 pair be used for such severe tolerances. We have done so here to help illustrate the effect of close approaches on the step-size.

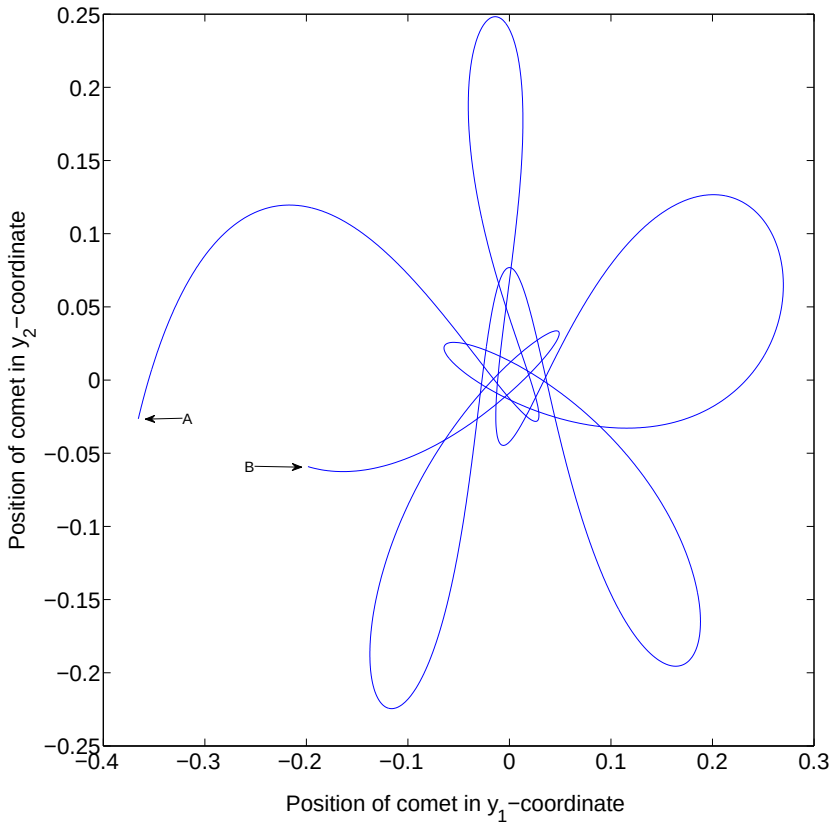

Figure 1.1: The phase-plane plot in the $y_1 - y_2$ plane of the position of the comet relative to Jupiter for the HRC Problem. The plot spans the time from $x = 1000$ days (A) to $x = 8000$ days (B).

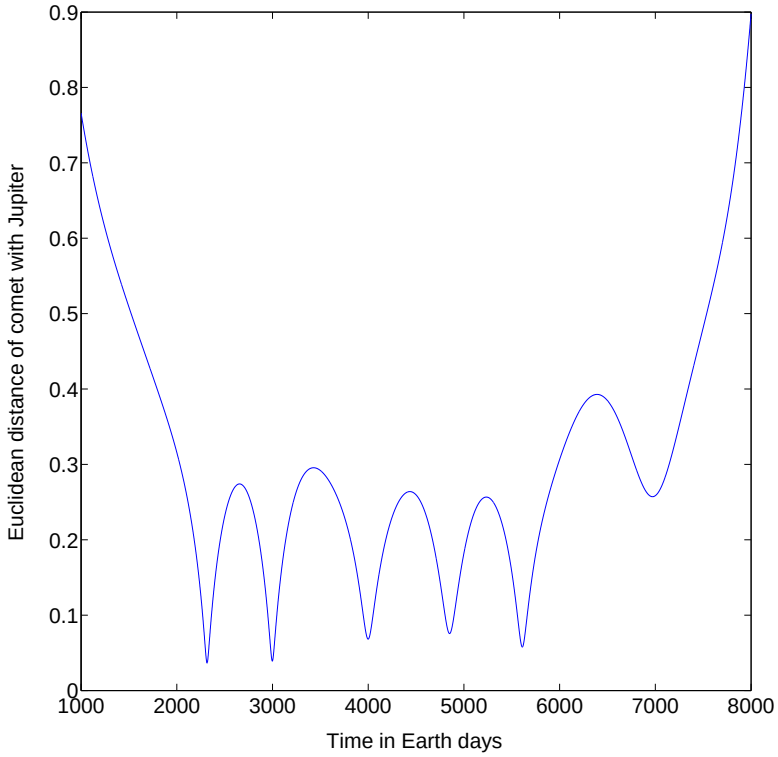

Figure 1.2: The distance from Jupiter to the comet in the HRC Problem. The comet makes five close approaches to Jupiter over approximately 4000 days, clearly shown in between 2000 and 6000 days.

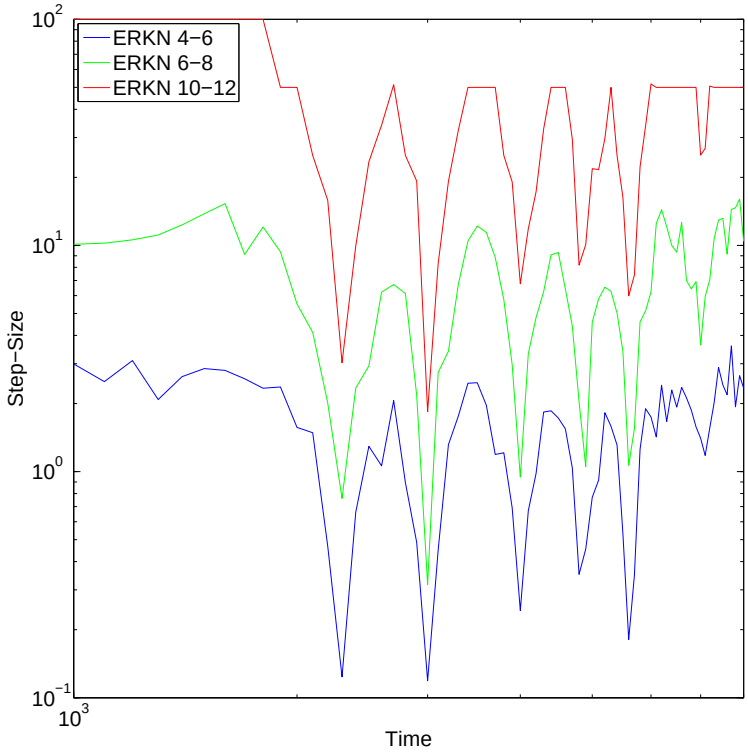

Figure 1.3: The average step-size versus time for the explicit Runge–Kutta Nyström pairs applied to the HRC Problem. The blue, green and red lines are for the 4-6, 6-8 and 10-12 pairs of Dormand [20, 21].

1.6 Saturnian Satellites Problem

To widen the diversity of the problems simulated, we include simulations of some Saturnian satellites. These satellites consist of Titan, Hyperion and Iapetus, the first and the last of which are the largest and third largest moons of Saturn, while Hyperion is the largest known irregular body in the Solar System. The eccentricities, masses and orbital periods of the Saturnian Satellites are listed in Table 1.2.

Sinclair and Taylor [84] used numerical integration to analyse the orbits of these satellites. The equations of motion included perturbations from the Sun and Saturn's inner satellite Rhea. It also involves terms modelling the effect of Saturn's oblateness. Let $\mathbf{r}_1$, $\mathbf{r}_2$, $\mathbf{r}_3$ and $\mathbf{r}_4$ denote the position of Titan, Hyperion, Iapetus and Rhea at time x, the coordinates being the Cartesian with origin at the center of mass of Saturn and its inner satellites (excluding Rhea). The equations of motion of the satellites for $i = 1, 2, 3$, are (the position of Rhea is discussed below)

$$\begin{aligned}\mathbf{r}_i'' = &-\frac{GM(1+m_i)\mathbf{r}_i}{\mathbf{r}_i^3} + \sum_{j=1, j\neq i}^{4} GMm_j\left(\frac{\mathbf{r}_j-\mathbf{r}_i}{\mathbf{r}_{ij}^3} - \frac{\mathbf{r}_j}{\mathbf{r}_j^3}\right)\\ &+ GM_s\left(\frac{\mathbf{r}_s-\mathbf{r}_i}{\mathbf{r}_{is}^3} - \frac{\mathbf{r}_s}{\mathbf{r}_s^3}\right) + \nabla_i R_i + \sum_{l=1}^{3} m_l \nabla_l R_l,\end{aligned} \tag{1.6.1}$$

where $\mathbf{r}_j$ is the position vector of the j^{th} satellite and m_j is the mass of the j^{th} satellite divided by the mass of Saturn (M), G is the gravitational constant, and M_s and $\mathbf{r}_s$ are the mass and position of the Sun respectively. The term $\nabla_i R_i$ corresponds to the effect of the oblateness of the Sun on the i^{th} satellite. The term $m_l \nabla_l R_l$ for $l = 1, 2, 3$, occurs due to the component of the attraction on Saturn caused by the oblateness of Saturn [84]. The term $\nabla_i R_i$ is

$$\nabla_i R_i = A\mathbf{r}_i + B\hat{\mathbf{z}},$$

where $\hat{\mathbf{z}}$ is the unit vector in the z-direction. The coefficients A and B are given as

$$A = \frac{GM}{\mathbf{r}_i^3}\sum_{n=2}^{4} J_n \frac{a_0^n}{\mathbf{r}_i^n} P'_{n+1}(z_i/\mathbf{r}_i), \qquad B = -\frac{GM}{\mathbf{r}_i^2}\sum_{n=2}^{4} J_n \frac{a_0^n}{\mathbf{r}_i^n} P'_{n}(z_i/\mathbf{r}_i),$$

where a_0 is the equatorial radius of Saturn, J_n is a non-dimensional constant and P_n is the Legendre polynomial of degree n.

Satellite	Orbital period (Days)	Eccentricity	Semi-major axis (AU)	Mass
Titan	15.9	0.028	0.0082	1350×10^{20}
Hyperion	21.3	0.123	0.0099	0.05×10^{20}
Iapetus	79.3	0.028	0.0238	18×10^{20}
Rhea	4.5	0.001	0.0035	23×10^{20}

Table 1.2: Some orbital and physical data for the Saturnian Satellites. The semi-major axis is in astronomical units (AU) and the mass in kilograms.

The position of Rhea in Cartesian coordinates is given by

$$\mathbf{r_4} = a\cos(L)\hat{i} + a\sin(L)\hat{j}$$

where $L = 231^\circ.761 + 79^\circ.69004007(x - 2411093.0)$ and a is a constant. The unit vectors $\hat{i}$ and $\hat{j}$ correspond to the $y_1$and y_2 coordinates of the vector $\mathbf{r}$.

We omit the Sun from our simulations, since they did not change the essential numerical properties of the problem, and the omission simplifies our testing.

1.7 Framework of Thesis

A large number of numerical methods for performing N-body simulations have been developed so far. These can be divided into two broad ways: those methods intended to obtain qualitative informations and those intended to obtain accurate solutions.

In this thesis, we present some new high order explicit Runge–Kutta Nyström methods. The new methods are up to approximately 60% more efficient than existing methods on our test problems. We also include some symplectic methods consisting of implicit Gauss methods and implemented in a more efficient way to reduce the computational cost by 20%. Throughout the thesis, we aim to analyse and compare the efficiency and error growth for new and existing methods. This error growth is examined in terms of global error and the errors in the Hamiltonian and angular momentum for these systems. We measure the L_2-norm of global error and relative error in energy and angular momentum throughout the comparisons.

Basic concepts, definitions and a review of traditional numerical methods for solv-

ing ordinary differential equations are developed in Chapter 2. Hamiltonian systems and their conserved quantities are also described in this chapter. In Chapter 3, we present the new high order explicit Runge–Kutta Nyström methods. A summary of extensive numerical comparisons of these methods with existing methods is also presented, when applied to a variety of Solar Systems problems. Then in Chapter 4, we review the implicit Runge–Kutta and Störmer methods that have optimal error growth. We also investigate a general way of improving the efficiency of the implicit Runge–Kutta methods and perform comparisons between the Störmer and implicit Runge–Kutta methods. Continuous extension has also been considered using interpolation polynomials for implicit Runge–Kutta methods. We end in Chapter 5 with our conclusions.

2

Preliminaries

There are two general ways to find the numerical solution of a second order IVP of the form (1.1.2). The first is to transform the problem to a system of first order equations and then perform the integration using one of a large array of methods including an Adams method, an extrapolation method or an explicit or implicit Runge–Kutta method. The second way is to solve the IVP directly using methods such as Störmer, Runge–Kutta Nyström, or extrapolation methods. In this chapter, we will survey some numerical methods that can be used to solve initial value ODE's that arise in N-body simulations of the Solar System.

2.1 First order systems

The equivalent first order IVP (1.1.2) takes the form

$$(y(x), u(x))' = (u(x), f(x, y(x))), \quad y(x_0) = y_0, \; u(x_0) = y_0'. \tag{2.1.1}$$

Before we look at the ways to numerically approximate the solution to an IVP, it is essential to consider whether there exists a solution to an initial value problem and if it exists, whether it is unique. These matters are discussed in the following.

2.1.1 Existence and uniqueness

There are many criteria for determining the existence and uniqueness of solutions. The most commonly used approach employs the Lipschitz condition, which is considered as a necessary condition for the existence of a unique solution to a system of differential equations and is given in the following theorem [8].

Theorem 2.1.1 *Let the function $f(x,y) : \mathbb{R}\times\mathbb{R}^n \longrightarrow \mathbb{R}^n$ be continuous for all $(x,y) \in D$, where D is defined by $a \leq x \leq b,\ -\infty \leq y_i \leq \infty,\ i = 1,\ldots,n$. If there exists a constant L then*

$$\|f(x,y) - f(x,y^*)\| \leq L\|y - y^*\|, \tag{2.1.2}$$

holds for all $(x,y),(x,y^*) \in D$, then for any $y_0 \in \mathbb{R}^n$ there exists a unique solution $y(x)$ of problem (2.1.1), where $y(x)$ is continuous and differentiable for all $(x,y) \in D$. The above condition (2.1.2) is called the Lipschitz condition and L the Lipschitz constant.

2.1.2 Order and convergence

Almost all numerical methods for first order IVP of the form (2.1.1) can be written using the increment formula

$$y_{n+1} = y_n + h_{n+1}\Phi(x_n, y_{n+1}, y_n, \ldots, y_{n-k+1}). \tag{2.1.3}$$

This yields a sequence of values y_n, $n = 0,1,\ldots,N$. The y_n are approximations to the true values $y(x_n)$ and $h_{n+1} = x_{n+1} - x_n$ is the step-size. The notation h_{n+1} is used instead of h, because it is possible to change the step-size at each step.

When using a numerical method to solve an IVP, we must make sure that these approximations satisfy certain conditions. One of them is convergence, that is as the

step-size h_{n+1} tends to zero, the numerical solution approaches the exact solution in the absence of round-off error. A numerical method defined by (2.1.3) is convergent if for all IVP satisfying Theorem 2.1.1

$$\max \|y(x_n) - y_n\| \longrightarrow 0 \quad \text{as} \quad h \longrightarrow 0,$$

with fixed value of x_n ($x_n = x_0 + nh$).

Another important property is how fast a numerical approximate solution converges to its exact solution. This can not be measured without the local truncation error. To understand this concept, we first define the local solution $z_n(x)$ at x_n as the solution of the local problem

$$z_n' = f(x, z_n), \quad z_n(x_n) = y_n.$$

The local truncation error is a measure of how much the numerical solution fails to satisfy the local problem. The local truncation error for the method (2.1.3) is defined as

$$t_{n+1} = y(x_n) + h_{n+1}\Phi(x_n, y(x_{n+1}), y(x_n), \ldots, y(x_{n-k+1})) - y(x_{n+1}).$$

We also need the concept of order. The order of a numerical method is measured by comparing the numerical solution on one step with the Taylor series expansion of the exact solution when written in the increment form

$$y(x_{n+1}) = y(x_n) + h\Delta(x, y(x_n)), \tag{2.1.4}$$

where

$$\Delta(x, y(x_n)) = y'(x_n) + \frac{h}{2}y''(x_n) + \cdots + \frac{h^{(p-1)}}{p!}y^{(p)}(x_n) + \cdots .$$

If the Taylor series expansion of the numerical solution (2.1.3) and exact solution agree to the terms up to the power of h^p, the method is of order p. The difference is of $O(h^{p+1})$, and is the local truncation error. As an example, the numerical solution y_{n+1} calculated using the Euler's method is given as

$$y_{n+1} = y_n + hf(x_n, y_n). \tag{2.1.5}$$

A comparison of equations (2.1.4) and (2.1.5) gives the local truncation error

$$y(x_{n+1}) - y_{n+1} = O(h^2).$$

The order is therefore one. In addition to the order of the method we are also interested to know how sensitive the solution is to small perturbations in the initial conditions. This concept is related to the stability of the numerical method.

2.1.3 Stability

Let ν_n^* be any perturbation in the initial condition ν_n of an IVP and y_n^* be the corresponding perturbed solution of y_n. Then if there exist constants K and ε such that

$$\|y_n - y_n^*\| \leq K\varepsilon,$$

whenever

$$\|\nu_n - \nu_n^*\| \leq \epsilon, \quad 0 \leq n \leq N,$$

then the method (2.1.3) is said to be stable. Otherwise, the method is said to be unstable. So a method is said to be stable if small changes in the input lead to small changes in the output. Stability does not mean that the numerical solution obtained by a method is accurate. Stability is also referred to as zero-stability in some texts, e.g. in [61].

Consider the linear test equation

$$y' = \lambda y, \quad y(0) = 1, \tag{2.1.6}$$

where $\lambda \in C$. The exact solution is $y(x) = e^{\lambda x}$, hence $\lim_{x\to\infty} y(x) = 0$ iff $Re\lambda < 0$.

Suppose the test equation is solved using the explicit Euler's method with a fixed step-size h. The approximate solution at $x_n = x_0 + nh$ is

$$y_n = (1 + h\lambda)y_{n-1},$$

or

$$y_n = (1 + h\lambda)^n y_0.$$

We want a bounded behaviour for $(1 + h\lambda)^n$ as $n \to \infty$ whenever the exact solution $\exp(nh\lambda)$ is bounded. This implies that $|1 + h\lambda| \leq 1$ iff $Re\lambda \leq 0$. We may take $z = h\lambda$, where z is complex. So the stability region for Euler's method is $z \in C$ satisfying $R(z) = |1 + z| \leq 1$, i.e. the disc with centre -1 and radius 1. If $Re(z) < 0$ and $|R(z)| \leq 1$, then

the method is absolutely stable. If $|R(z)| = 1$ and λ is purely imaginary, the method is said to be periodically stable or P-stable.

To extend the stability analysis to a system of differential equations, we consider a system of linear differential equations of dimension m

$$y' = Ay,$$

where A is an $m \times m$ constant diagonalisable matrix. Euler's method applied to the above equation, gives the solution

$$y_n = (I + hA)^n y_0.$$

The exact solution $y(x_n) = \exp(nhA)y(x_0)$ of the above system is stable iff $Re\lambda_i \leq 0$ for all $i = 1, \ldots, m$, where the λ_i are the eigenvalues of A. So we deduce that the step-size $h > 0$ must be such that $|1 + h\lambda_i| < 1$ and all the products $h\lambda_1, h\lambda_2, \ldots, h\lambda_m$ lie in the stability region.

2.1.4 Local and global error

Two measures of the accuracy of a numerical method are the local and global errors. We define the local error at x_{n+1}. This error arises from a single step and is

$$\mathcal{E}_{\text{le}} = y_{n+1} - z_n(x_{n+1}), \quad z_n = y_n.$$

The error is closely related to the local truncation error. The difference between the exact and numerical solution is said to be the global error of the numerical solution and is defined as

$$\mathcal{E}_{\text{ge}} = y_{n+1} - y(x_{n+1}).$$

The global error is of major importance in the measurement of the quality of the approximated solution at time x_{n+1}.

2.1.5 Round-off error

When performing accurate simulations, a significant contribution to the global error is the round-off error. Since computers store numbers to only a certain precision, there will be a loss of accuracy when a long-term computation is involved, especially when using a small step-size. We will illustrate and measure the effects of round-off in subsequent chapters.

2.2 Numerical integrators

Initial value problems can be divided into stiff and non-stiff. There does not exist a formal definition of stiffness. Stiffness was first recognised in 1952 by Curtiss and Hirschfelder [17]. One feature of stiff problems is that they normally have a large Lipschitz constant. Explicit methods are not useful for solving stiff problems as the methods have bounded stability regions, necessitating excessively small step-sizes.

Most IVPs that arise when performing N-body simulations of the Solar System are non-stiff. For the remainder of the thesis, we will assume non-stiff problems.

2.2.1 Multi-step integrators

Linear multistep methods use the values of the solution and derivatives from the previous steps. The general form of a k-step linear multistep method for first order system is given by

$$y_n = \sum_{j=1}^{k} \alpha_j y_{n-j} + h \sum_{j=0}^{k} \beta_j f(y_{n-j}), \qquad k = 1, 2, \ldots, \tag{2.2.1}$$

where α_j and β_j are given constants, h denotes the step-size and y_r is the numerical approximation to the exact value $y(x_r)$ at the point x_r. For $k > 1$, a special procedure must be used to find the starting values $y_1, \ldots, y_{k-1}$.

The method (2.2.1) is said to be explicit if $\beta_0 = 0$ and implicit if $\beta_0 \neq 0$. The Adams–

Bashforth and Adams–Moulton are special cases of linear multistep methods which are explicit and implicit respectively and intended for non-stiff problems. Adams–Moulton methods use less information from the past compared with the Adams–Bashforth method while obtaining the same accuracy. Another notable thing is that the coefficients for Adams–Moulton methods are smaller than that of Adams–Bashforth.

Adams methods were first used as predictor-corrector pairs by Milne [67]. On each step, the Adams-Bashforth formula was first used to predict y_n. The Adams-Moulton formula was then used one or more times to correct the predicted value. The predictor-corrector mode is often implemented as $P(EC)^mE$ or $P(EC)^m$, where m is the number of iterations for the corrector formula and E means an evaluation of f. This implementation has two advantages. It is explicit in nature and the local error can be estimated using a simple technique known as Milne's device [61].

Linear multistep methods are often implemented in variable step-size and variable-order fashion, to produce an adaptive code. The first efficient adaptive code was published by Krogh [59].

An important class of multistep methods for solving a second order system of equations is Störmer methods. These methods are popular in astronomical applications and have long been used for long-term simulations of the Solar System [39]. Störmer [86] introduced these methods. Störmer developed a simple method by adding the Taylor series for $y(x_n + h)$ and $y(x_n - h)$, ignoring the higher order terms as detailed in [47], and obtained

$$y_{n+1} - 2y_n + y_{n-1} = h^2 f_n.$$

Higher order is obtained by using differences involving values of f from the end of the previous steps, for example

$$y_{n+1} - 2y_n + y_{n-1} = h^2 f_n + \frac{h^2}{12}\left(\frac{59}{20}f_n - \frac{176}{20}f_{n-1} + \frac{194}{20}f_{n-2} - \frac{96}{20}f_{n-3} + \frac{19}{20}f_{n-4}\right).$$

A Störmer method of order $p+1$ can be written in the form

$$y_{n+1} - 2y_n + y_{n-1} = h^2 \sum_{i=0}^{p-1} \alpha_i f_{n-i}, \tag{2.2.2}$$

$$y'_{n+1} - \frac{1}{h}(y_n - y_{n-1}) = h\sum_{i=0}^{p-1} \beta_i f_{n-i}.$$

The coefficients α_i and β_i can be found from generating functions [47]. The starting values $y_1, y_2, \ldots, y_{p-1}$ are usually computed using a one-step method. The detailed implementation of Störmer methods is discussed in Chapter 4.

2.2.2 One-step integrators

Integrators which use only information about the solution at a single point y_n to integrate forward to the next point y_{n+1} are known as one-step integrators. One of the earliest numerical algorithms for the approximation of the solution of an IVP is the Taylor series method. The method advances the step by calculating higher derivatives. A p^{th} order method is written in increment form as

$$y_{n+1} = y_n + h\Phi(x, y_n), \tag{2.2.3}$$

where

$$\Phi(x, y_n) = y'_n + \frac{h}{2}y''_n + \cdots + \frac{h^{(p-1)}}{p!}y_n^{(p)}.$$

The relative difficulty of computing higher derivatives initially restricted the popularity of the Taylor series methods. The methods have become more popular with the advent of automatic differentiation and canonical transformation. To avoid the difficulty of computing higher derivatives, Runge–Kutta (RK) methods were devised about a century ago. The German mathematician Runge [75] and his successors Heun [49] and Kutta [60] established some fundamentals of RK methods during the late 19^{th} and early 20^{th} centuries. Kutta also characterized the most famous "Classical RK method" of order four. All these methods are explicit in nature so they are easy to implement. The basic approach of RK methods is to obtain the Taylor series expansion for the exact and approximate solutions without evaluating the derivatives of f and comparing these series term by term at the end of each single step.

In the 1960's Kuntzmann and Butcher [7] proposed implicit Runge–Kutta (IRK) methods. Since that time considerable attention has been devoted towards improvement in the efficiency and cheaper implementation of these methods. A number of interesting subclasses of the IRK methods have been identified. These methods represent attempts to

trade-off the higher accuracy of the IRK methods for methods which can be implemented more efficiently. Such methods include semi-implicit methods (SIRK) and the diagonally implicit Runge–Kutta methods (DIRK) introduced by Alexander in [2].

Butcher [7] introduced methods based on quadrature formula. These are known as Gauss, Radau and Lobatto methods. The Gauss methods have the maximum possible order $2s$ (s is the number of stages of the method) and have strong stability properties. The Radau methods are of order $2s-1$ and Lobatto methods of order $2s-2$. The implementation of Gauss IRK methods is discussed in Chapter 4.

For an ordinary differential equation of type (2.1.1), a general Runge–Kutta method is of the form

$$y_{n+1} = y_n + h\Phi(y_n), \tag{2.2.4}$$

where

$$\Phi(y_n) = \sum_{i=1}^{s} b_i K_i,$$

and

$$K_i = f(x_n + c_i h, y_n + h\sum_{j=1}^{s} a_{ij} K_j), \quad i = 1, 2, \ldots, s.$$

The $K_i,\ i = 1, \ldots, s$, are called stages and are calculated during the integration from x_n to x_{n+1}. As with methods discussed previously, the output value y_{n+1} is the numerical approximation to the true solution y at $x = x_{n+1}$. The coefficients of the methods are often written in a form known as the Butcher tableau shown in Table 2.1,

c_1	a_{11}	a_{12}	$\ldots$	a_{1s}
c_2	a_{21}	a_{21}	$\ldots$	a_{2s}
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
c_s	a_{s1}	a_{s2}	$\ldots$	a_{ss}
	b_1	b_2	$\ldots$	b_s

Table 2.1: The Butcher tableau for RK methods.

The b_i are called the exterior weights of the method and the c_i are the abscissas. The row sum condition $c_i = \sum_{j=1}^{s} a_{ij}, \quad i = 1, 2, \ldots, s$ holds, for all but some low

order methods. The consistency condition $\sum_{i=1}^{s} b_i = 1$ always holds. The $s \times s$ matrix $A = (a_{ij})_{i,j=1}^{s}$ is called the RK matrix and the elements of matrix A are known as interior weights. The method is explicit if matrix A is strictly lower triangular and implicit otherwise. As noted previously, implementation of implicit methods is less convenient than that of explicit Runge–Kutta methods because at each stage the values of vector K_i depend upon other K_j. So at each step of computation a set of $n \times s$ (n be the dimension of system) non-linear system of equations is to be solved.

The Runge–Kutta methods for solving second order differential equations of the form (1.1.1) directly are known as Runge–Kutta Nyström methods (RKN). These were introduced in 1925 by E. J. Nyström [68]. A RKN method can be written as [19]

$$\begin{aligned} y_{n+1} &= y_n + hy'_n + h^2 \sum_i b_i k_i, \\ y'_{n+1} &= y'_n + h \sum_i b'_i k_i, \end{aligned} \tag{2.2.5}$$

and

$$k_i = f(x_n + c_i h, y_n + c_i h y'_n + h^2 \sum_j a_{ij} k_j).$$

The Butcher tableau for the method is shown in Table 2.2

c_1	a_{11}	a_{12}	$\dots$	a_{1s}
c_2	a_{21}	a_{21}	$\dots$	a_{2s}
$\vdots$	$\vdots$			$\vdots$
c_s	a_{s1}	a_{s2}	$\dots$	a_{ss}
	b_1	b_2	$\dots$	b_s
	b'_1	b'_2	$\dots$	b'_s

Table 2.2: The Butcher tableau for RKN methods.

RKN methods are relatively simple to apply and reduce the computational work compared with RK methods applied to the equivalent first order problem. As an example, an order-five explicit RKN method needs only four function evaluations while an explicit Runge–Kutta method of order five needs at least six function evaluations [47].

The coefficients of the above methods can be found by solving the order conditions

for that method. These can be derived in a similar way as for Runge–Kutta methods, i.e. by comparing the Taylor series expansions for the exact and numerical solutions.

2.2.3 Adaptive step-size methods

The efficiency of numerical methods for the approximate solution of ordinary differential equations depends on the strategy for controlling the error in the approximate solutions. One procedure is to use adaptive step-size control to achieve a predetermined accuracy on each step with minimal computational effort. In order to control the error, a pair of formulae of different orders is used at one-step so that the derivative evaluations of the two methods are identical. The essence of this idea was first introduced by Merson [66], and further developed by England [27] and Fehlberg [31]. In embedded Runge–Kutta methods, two methods of different orders p and q are used with the same set of stage vectors K_i and can be written as

$$\begin{aligned} \hat{y}_{n+1} &= \hat{y}_n + h\sum_{i=1}^{s} \hat{b}_i K_i, \\ y_{n+1} &= \hat{y}_n + h\sum_{i=1}^{s} b_i K_i, \end{aligned} \tag{2.2.6}$$

where

$$K_i = f(x_n + c_i h,\ \hat{y}_n + h\sum_{j=1}^{s} a_{ij} K_j), \quad i = 1, 2, \ldots, s.$$

A pair of formulae can be represented in a Butcher tableau as

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

where $b^T = [b_1, \ldots, b_s]^T$ is the exterior weight vector of the RK method of order p and $\hat{b}^T = [\hat{b}_1, \ldots, \hat{b}_s]^T$ is that of order q, $q > p$. A pair of formulae having orders p and q is usually referred to as a $p - q$ pair. If the solution y_i obtained by the p^{th} order method is used as the starting value for the step continuation then the embedded pair is said to be implemented in lower order fashion. However for efficiency reasons, it is recommended

that the solution y_i obtained by the q^{th} order method should be used for the next step [28]. In this way, the more accurate approximation is used to advance the integration.

The pair operated in this way is said to be implemented in higher mode or local extrapolation. The difference between the order p and q methods will give an estimate of the local truncation error $e_{n+1} = \hat{y}_{n+1} - y_{n+1}$ at x_{n+1}. Once the local error e_{n+1} of an approximate solution is known, then using a method of order p it is easy to control the step-size by using the well-known formula [56]

$$h_{n+1} = 0.9h_n \left(\frac{\text{Tol}}{\|e_{n+1}\|} \right)^{\frac{1}{p+1}}, \tag{2.2.7}$$

where Tol is the local error tolerance. During the integration a new step will be accepted if $\|e_{n+1}\| \leq$ Tol and the step-size for the next step will be calculated using (2.2.7). If $\|e_{n+1}\| >$ Tol, the step will be rejected and a reduced step-size will be calculated using the same formula.

Dormand and Prince [24] popularised the idea of a method having the property FSAL (first same as last), in which the vector b^T and the last row of the matrix A are equal. A method developed by Dormand and Prince [22] in 1980 known as DOPRI (5,4) has seven stages but, being FSAL, the method has effectively six stages because the last stage is reused as the first stage of the next step.

In a similar way to the embedding technique for RK methods, a RKN algorithm utilises an estimate of the local truncation error of both y and y' using two pair having order p and q and sharing the same function evaluations

$$\begin{aligned} \hat{y}_{n+1} &= \hat{u}_n + h_n \hat{u}'_n + h_n^2 \sum_i^s \hat{b}_i k_i, \\ \hat{y}'_{n+1} &= \hat{u}'_n + h_n \sum_i^s \hat{b}'_i k_i, \end{aligned} \tag{2.2.8}$$

$$\begin{aligned} y_{n+1} &= u_n + h_n u'_n + h_n^2 \sum_i^s b_i k_i, \\ y'_{n+1} &= u'_n + h_n \sum_i^s b'_i k_i, \end{aligned} \tag{2.2.9}$$

where

$$k_i = f(x_n + c_i h_n, \hat{y}_n + c_i h_n \hat{y'}_n + h_n^2 \sum_j a_{ij} k_j), \quad i = 1, \ldots, s.$$

The caps denote the approximations for the q^{th} order method. If the numerical approximations are taken from x_n to x_{n+1} using the q^{th} order formula then $u_n = \hat{y}_n$, $u'_n = \hat{y}'_n$. If the p^{th} order method is used to advance the step then $u_n = y_n$, $u'_n = y'_n$. It is practically preferable to implement the pairs in higher order mode [51]. Throughout the thesis, we implement the pairs in local extrapolation mode and denote explicit Runge–Kutta Nyström (ERKN) pairs as ERKN $p - q$ pair.

Fehlberg [30, 32] was the first who developed RKN pairs. Later Dormand and Prince [24], Bettis [5], Horn [51] and Filippi and Graf [34] also added their algorithms in the RKN family. Many classes of fully implicit, diagonally implicit and explicit embedded RKN methods have been developed so far. Such methods can be seen in Dormand *et al.* [20, 21], Someijer [85], Sharp *et al.* [82], Papageorgiou *et al.* [70], El-Mikkawy *et al.* [25] and Al-Khasawneh *et al.* [1].

The local error for ERKN method is computed using a similar expression as (2.2.7)

$$h_{n+1} = 0.9 h_n \left(\frac{\text{Tol}}{\max\{\|e_{n+1}\|, \|e'_{n+1}\|\}} \right)^{\frac{1}{p+1}}, \tag{2.2.10}$$

where $e_{n+1} = \hat{y}_{n+1} - y_{n+1}$ and $e'_{n+1} = \hat{y}'_{n+1} - y'_{n+1}$ are local error estimates in the p^{th} order formula.

As noted previously linear multistep methods including Störmer methods can be implemented with a variable step-size, see for example Cano and Archilla [11]. In this thesis we will be using fixed step-size multistep methods, as discussed in subsequent chapters.

2.3 Hamiltonian Systems

The equations of motion for many N-body simulations not only contain the information about position and velocities but some hidden geometrical properties. These geometrical properties include phase space, symmetries of the motion and some special conservation laws for the energy, angular momentum and centre of mass. The branch of physics, which deals with these physical laws is called classical mechanics. A form of classical mechanics in which equations of motion are based on generalised coordinates q_i and generalised momenta p_i is called as Hamiltonian mechanics. A system governed by these equations of motion is called a Hamiltonian system. These systems arise on a large scale in cosmology and dynamical astronomy and on a small scale in molecular dynamics.

The Lagrange theory of dynamical systems is related to Hamiltonian mechanics and based on real valued functions. These are the kinetic energy $T(p)$ and the potential energy $V(q)$. The Lagrangian is defined as $L = T - V$, and the Lagrangian equations of motion are given by

$$\frac{\partial L}{\partial q} = \frac{d}{dx}\left(\frac{\partial L}{\partial q'}\right). \tag{2.3.1}$$

The Hamiltonian equations of motion come from Lagrange's equations and are written in autonomous form as [78]

$$\frac{dq_i}{dx} = \frac{\partial H(q,p)}{\partial p_i}, \qquad \frac{dp_i}{dx} = -\frac{\partial H(q,p)}{\partial q_i}, \tag{2.3.2}$$

for $i = 1, \ldots, n$, and $H : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$. The function H is called the Hamiltonian. In the Hamiltonian equations, the number of (q_i, p_i) pairs is said to be the number of degrees of freedom of that system.

For many dynamical problems the Hamiltonian is in separable form, since the Hamiltonian H is the sum of the kinetic energy and the potential energy of the system. Hence

$$H(q,p) = T(p) + V(q), \tag{2.3.3}$$

where T is normally quadratic in p, *i.e.* $T(p) = \frac{1}{2}p^T p$.

If H does not depend explicitly on time x, the Hamiltonian equations are autonomous and describe a conservative system. In a conservative systems, the Hamiltonian

function H is a first integral of (2.3.2). To see this form dH/dx, this is

$$\frac{dH}{dx} = \frac{\partial H}{\partial x} + \sum_{i=1}^{n} \frac{\partial H}{\partial q_i} q_i' + \sum_{i=1}^{n} \frac{\partial H}{\partial p_i} p_i'.$$

Now use equation (2.3.2) and that $\partial H/\partial x = 0$. We have,

$$\frac{dH}{dx} = \sum_{i=1}^{n} \frac{\partial H}{\partial q_i}\frac{\partial H}{\partial p_i} + \sum_{i=1}^{n} \frac{\partial H}{\partial p_i}\left(-\frac{\partial H}{\partial q_i}\right) = 0.$$

This means that the Hamiltonian is conserved for an autonomous Hamiltonian system. In most cases, the first integral can be identified with the energy of the system. Its invariance corresponds to the conservation of total energy.

Sometimes, it is useful to re-write (2.3.2) by defining $y = (q_1, \ldots, q_n, p_1, \ldots, p_n)$ as a $2n$-dimensional vector. Then (2.3.2) takes the form

$$\frac{dy}{dx} = J^{-1}\nabla H, \tag{2.3.4}$$

where the $2n \times 2n$ skew symmetric matrix J is defined as

$$J = \begin{bmatrix} 0 & I \\ -I & 0 \end{bmatrix}, \tag{2.3.5}$$

and I is the $n \times n$ identity matrix.

We now present two simple examples of Hamiltonian systems. More realistic examples are described in subsequent chapters.

Example 2.3.1 *The harmonic oscillator*

An example of a Hamiltonian system is a mass-spring system modeled as a simple harmonic oscillator having kinetic energy $p^2/(2m)$, where $p = mv$ is the momentum of the system, and potential energy $\frac{1}{2}kq^2$, where k is the spring constant. The parameter q is the distance of the body of mass m from the equilibrium. The Hamiltonian is the total energy of the system and has one degree of freedom

$$H(q,p) = \frac{1}{2}kq^2 + \frac{p^2}{2m}. \tag{2.3.6}$$

In the case where $k = m = 1$, the equations of motion from the Hamiltonian are

$$q' = \frac{\partial H(q,p)}{\partial p} = p, \qquad p' = -\frac{\partial H(q,p)}{\partial q} = -q. \tag{2.3.7}$$

Example 2.3.2 *The simple pendulum*

A simple pendulum consists of a bob of mass m at the end of a massless string of length l. A simple pendulum has one degree of freedom and the Hamiltonian can be written as

$$H(q,p) = \frac{p^2}{2m} - mgl\cos(q), \tag{2.3.8}$$

where g is the acceleration due to gravity. The equations of motion are

$$q' = \frac{\partial H(q,p)}{\partial p} = \frac{p}{m}, \qquad p' = -\frac{\partial H(q,p)}{\partial q} = -mgl\sin(q). \tag{2.3.9}$$

2.3.1 Symplecticity

The main qualitative property of many Hamiltonian systems is the preservation of phase flow of the underlying symplectic structure in phase space. This is called symplecticity. The phase space of Hamiltonian systems is a $2n-$dimensional space in (p, q). One question that arises is why we need to preserve this qualitative property of symplectic structure? Since Hamiltonian systems naturally have these properties, it is beneficial to preserve them numerically as well. Standard methods for simulating motion do not explicitly attempt to satisfy the physical laws which are intrinsic to Hamiltonian systems.

The phase flow of Hamiltonian systems using the operator Φ_H is a transformation such that

$$\Phi_H : (p_0, q_0) \longmapsto (p(x), q(x)).$$

The transformation Φ_H is a symplectic or canonical transformation according to Liouville's theorem: the phase flow preserves area, an important property of (2.3.2). In term of differential forms, the corresponding flow is symplectic and preserves the differential 2-form

$$w = \sum_{i=1}^{n} dp_i \wedge dq_i,$$

In two dimensions ($n = 1$), this means that the area of parallelograms at different times remains unchanged as shown in the following figure.

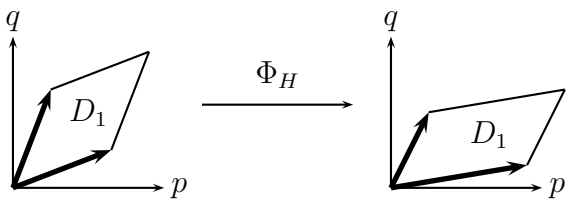

Figure 2.1: Illustration of symplecticity.

2.3.2 Symplectic integrators

The vital property of Hamiltonian systems is that the Hamiltonian $H(p, q)$ is a first integral of system (2.3.2). The numerical methods which preserve the so-called symplectic structure of the variables (p, q) at each step during numerical integration, *i.e.* reproducing the qualitative properties of the solution for the Hamiltonian systems, are called symplectic integrators. When solving the Hamiltonian systems, we can verify that certain numerical methods are symplectic by using the following definitions.

Definition 2.3.1 *A linear mapping $\Phi_H : R^{2n} \to R^{2n}$ is called symplectic if*

$$\Phi_H^T J \Phi_H = J,$$

where $J = \begin{bmatrix} 0 & I \\ -I & 0 \end{bmatrix}$.

Definition 2.3.2 *A differentiable map $g : U \longrightarrow R^{2n}$ (where $U \subset R^{2n}$ is an open set) is called symplectic if the Jacobian matrix $g'(p, q)$ is everywhere symplectic, i.e.,*

$$g'(p, q)^T J g'(p, q) = J.$$

Definition 2.3.3 *Let D be a domain in R^{2n} having the symplectic structure. A numerical one-step method consisting of a function $\phi_h : D \longrightarrow D$ with fixed $h > 0$ is called symplectic*

if the approximate solution can be computed as

$$(q_{n+1}, p_{n+1}) = \phi_h(q_n, p_n),$$

whenever the method is applied to a smooth Hamiltonian system.

Pioneering work on symplectic integrators is due to de Vogelaere [18], Ruth [76] and Feng [33]. Lasagni [62], Sanz-Serna [77] and Suris [87] independently found a condition for implicit Runge–Kutta methods to be symplectic. For details see Sanz-Serna and Calvo [78], and Hairer *et al.* [44]. All multistep methods are non-symplectic as they require more than one initial value to start. These methods can not define a map on the phase space and so can not be symplectic. One-step methods have the potential of being symplectic integrators. The family of explicit Runge–Kutta methods is non-symplectic as they introduce artificial dissipation during step by step integration. Also in general, they are unable to keep the Hamiltonian constant throughout the integration. For a RK method to be symplectic, its coefficients must satisfy the following theorem along with its order conditions.

Theorem 2.3.1 *Let $M = (m_{ij})_{i,j=1}^{s}$ be the real $s \times s$ matrix given by*

$$m_{ij} = b_i a_{ij} + b_j a_{ji} - b_i b_j,$$

for $i, j = 1, \ldots, s$. If the coefficients of a Runge–Kutta method satisfy $M = 0$, then it is symplectic.

The above matrix M arises frequently in the study of RK schemes and nonlinear stability and many implicit Runge–Kutta methods satisfy the condition and are symplectic. Some of the examples include the symplectic Euler, the implicit midpoint rule, the Störmer-Verlet and Gauss methods.

It is worth mentioning that the use of variable step-size can lead to non-symplectic behaviour; at each integration step the method should ensure that the underlying physical properties are preserved.

Embedded RKN methods with variable step-size are not symplectic while the fixed step-size explicit RKN methods developed by Yoshida [95], Okunbor and Skeel [69] and

Calvo and Sanz-Serna [9, 10] are symplectic. Similar to RK methods, a RKN method is symplectic if it satisfies

$$a_{ij} = (c_i - c_j)b'_j, \quad j = 1, \ldots, i-1, \;\; i = 2, \ldots, s.$$

A remarkable property of symplectic integrators other than the area preserving property is that the accumulation of discretization error in energy does not have a secular term. This means that the error in the conserved quantities does not grow with time if the round-off error is insignificant and the error in position increases linearly with time. Non-symplectic methods produce errors in the position and conserved quantities that grow quadratically in time.

3

Explicit Runge–Kutta Nyström Methods

In this chapter we review the construction of explicit Runge–Kutta Nyström methods and present new 8-10 and 10-12 Nyström pairs that are more efficient than those of El-Mikkawy [26] and Dormand *et al.* [21].

3.1 RKN embedded pairs

Let $\hat{y}_{n+1}$, $\hat{y'}_{n+1}$ be the numerical approximations of order q to $y(x_{n+1})$ and $y'(x_{n+1})$ respectively, and the y_{n+1}, y'_{n+1} the corresponding approximations of order p, where $p < q$. The update formulae (2.2.8) and (2.2.9) for an embedded RKN pair of s stages were introduced in the previous chapter. Here we use the more general formulation

$$\begin{aligned} \hat{y}_{n+1} &= \hat{u}_n + h_n \hat{\Phi}_n(x_n, \hat{y}_n, \hat{y}'_n, h_n), \\ \hat{y}'_{n+1} &= \hat{u}'_n + h_n \hat{\Phi'}_n(x_n, \hat{y}_n, \hat{y}'_n, h_n), \end{aligned} \tag{3.1.1}$$

$$\begin{aligned} y_{n+1} &= u_n + h_n \Phi_n, (x_n, \hat{y}_n, \hat{y}'_n, h_n), \\ y'_{n+1} &= u'_n + h_n \Phi'_n (x_n, \hat{y}_n, \hat{y}'_n, h_n), \end{aligned} \tag{3.1.2}$$

where

$$\begin{aligned} \hat{\Phi}(x_n, \hat{y}_n, \hat{y}'_n, h_n) &= \hat{u}'_n + h_n \sum_i^s \hat{b}_i k_i, \\ \hat{\Phi}'(x_n, \hat{y}_n, \hat{y}'_n, h_n) &= \sum_i^s \hat{b}'_i k_i, \\ \Phi(x_n, \hat{y}_n, \hat{y}'_n, h_n) &= u'_n + h_n \sum_i^s b_i k_i, \\ \Phi'(x_n, \hat{y}_n, \hat{y}'_n, h_n) &= \sum_i^s b'_i k_i, \end{aligned}$$

and

$$k_i = f(x_n + c_i h_n, \hat{u}_n + c_i h_n \hat{u'}_n + h_n^2 \sum_{j=1}^s a_{ij} k_j), \quad i = 1, \ldots, s.$$

We have assumed the local extrapolation mode (the solution is advanced using the high order formula).

Many classes of explicit embedded RKN methods have been developed. These methods include those of Dormand *et al.* [20, 21], Someijer [85], Papageorgiou *et al.* [70], El-Mikkawy *et al.* [25] and Al-Khasawneh *et al.* [1]. The numerical test results in the cited references show that embedded RKN methods require fewer function evaluations to achieve the same global error than embedded explicit Runge-Kutta methods applied to the equivalent first order system. RKN methods also require less storage.

The coefficients $a_{ij},\ \hat{b}_i,\ \hat{b}'_i,\ b_i,\ b'_i$ are usually chosen so that the method has the highest possible order. The coefficients a_{ij} are related to the abscissas c_i by

$$\sum_{j=1}^{i-1} a_{ij} = \frac{c_i^2}{2}, \quad i = 1, \ldots, s,$$

for all but some low order methods. The convergence and sufficient condition for a RKN method to be consistent are given by the following definitions [26].

Definition 3.1.1 *The RKN methods defined by (3.1.1) and (3.1.2) are said to be convergent for the problem (1.1.1) if for all fixed x_n,*

$$\max\left(|| \varepsilon_{\mathrm{ge}_n}||, || \varepsilon'_{\mathrm{ge}_n}||\right) \| \longrightarrow 0 \quad as \quad h \longrightarrow 0,$$

where $\varepsilon_{\mathrm{ge}}$ and $\varepsilon'_{\mathrm{ge}}$ are the global errors for y and y', as defined in section 2.1.4.

Definition 3.1.2 *The RKN methods defined by (3.1.1) and (3.1.2) are said to be consistent for the problem (1.1.1) if*

$$\Phi(y, y', 0) = y',$$

and

$$\Phi'(y, y', 0) = f(y).$$

The above definitions yield that the RKN methods (3.1.1) and (3.1.2) are consistent if

$$\sum_{i=1}^{s} \hat{b}' = 1 \quad \text{and} \quad \sum_{i=1}^{s} b' = 1.$$

3.1.1 Derivation of ERKN pair

As with ERK methods, the order conditions for ERKN methods are found by comparing the exact and numerical solution of a Taylor series expansion. This is illustrated in the following example:

Example 3.1.1

Using the Taylor series expansion, we have

$$\begin{aligned} y(x_{n+1}) &= y(x_n) + hy'(x_n) + \frac{1}{2}h^2y''(x_n) + \frac{1}{6}h^3y'''(x_n) + \frac{1}{24}h^4y^{(4)}(x_n) + O(h^5), \\ y'(x_{n+1}) &= y'(x_n) + hy''(x_n) + \frac{1}{2}h^2y'''(x_n) + \frac{1}{6}h^3y^4(x_n) + \frac{1}{24}h^4y^{(5)}(x_n) + O(h^5), \end{aligned}$$

where

$$\begin{aligned}
y' &= y', \\
y'' &= f(x,y), \\
y''' &= f_x + f_y y', \\
y^{(4)} &= f_{xx} + 2f_{xy}y' + f_y f + f_{yy}y'^2, \\
y^{(5)} &= f_{xxx} + 3f_{xxy}y' + 3f_{xy}f + 3f_{xyy}y'^2 + 5f_{yy}fy' + f_x f_y + f_y^2 y' + f_{yyy}y'^3.
\end{aligned}$$

The quantities $f(x,y)$ and its partial derivatives are called elementary differentials $D^i_j(x,y)$ (i is the order and j is number of the elementary differential). Assuming $s = 3$ and expanding the approximation (3.1.2) in the same manner yields

$$\begin{aligned}
y_{n+1} &= y_n + hy'(x_n) + h^2 f(b_1 + b_2 + b_3) + h^3(b_2c_2 + b_3c_3)(f_x + f_y y') \\
&\quad + h^4(b_2a_{21}ff_y + \left(\frac{1}{2}b_2c_2^2 + \frac{1}{2}b_3c_3^2\right)(f_{xx} + f_{yy}y'^2) + (b_2c_2^2 + b_3c_3^2)f_{xy}y') \\
y'_{n+1} &= y'_n + hf(b'_1 + b'_2 + b'_3) + h^2 f(b'_2c_2 + b'_3c_3)(f_x + f_y y') + h^3((f_{xx} + f_{yy}y'^2), \\
&\quad \left(\frac{1}{2}b_2c_2^2 + \frac{1}{2}b_3c_3^2\right) + b_2a_{21}ff_y + (b_2c_2^2 + b_3c_3^2)f_{xy}y') + h^4(a_{32}c_2f_{xxx} + a_{32}a_{21}f_x f_y \\
&\quad (3f_{xxy}y' + 3f_{xy}y'^2)\left(\frac{1}{2}b_2c_2^3 + \frac{1}{2}b_3c_3^3\right) + 3f_{xy}fy'(b_2c_2^2 + b_3c_3^2) + (f_y^2 y' + f_{yyy}y'^3)b_2a_{21}).
\end{aligned}$$

Now comparing the above two equations with the Taylor series expansion, we get the following equations for y

$$\begin{aligned}
b_1 + b_2 + b_3 &= \frac{1}{2}, \\
b_2c_2 + b_3c_3 &= \frac{1}{6}, \\
b_2c_2^2 + b_3c_3^2 &= \frac{1}{12},
\end{aligned}$$

and for the derivatives y'

$$\begin{aligned} b'_1 + b'_2 + b'_3 &= 1, \\ b'_2 c_2 + b'_3 c_3 &= \frac{1}{2}, \\ b'_2 c_2^2 + b'_3 c_3^2 &= \frac{1}{3}, \\ b'_2 c_2^3 + b'_3 c_3^3 &= \frac{1}{4}, \\ b'_3 a_{32} c_2 &= \frac{1}{24}. \end{aligned}$$

These equations are solved for the coefficients to obtain an ERKN embedded pair. Since, we have five equations for derivative order conditions and six unknowns, so one of the coefficients can be chosen as free parameter. It is convenient to have c_2 as free parameter.

We also need the order conditions for the formulae of order q. These involve the coefficients c_i, a_{ij}, $\hat{b}'_i$ and $\hat{b}_i$, and are found in a similar way as for the order p formulae.

The order conditions for the order p solution formula can be eliminated using the transformation

$$b_i = b'_i(1 - c_i), \qquad i = 1, \ldots, s. \tag{3.1.3}$$

Using this transformation, simplifies the solving of the order conditions but can mean that the solution to the order conditions is not as general.

The complexity of the order conditions grows rapidly as the order of the method increases. These order conditions can be represented in the form of trees and this notation is very useful in finding order conditions for higher orders. The Nyström trees for $y'' = f(y, y')$ are first given by Hairer and Wanner [48]. Nyström trees are described in details by Hairer *et al.* [47]

3.1.2 Simplifying assumptions

The complexity of the order conditions can be reduced by assuming specific relationships between coefficients. The relationships are called simplifying assumptions. One set of simplifying assumptions often used in the construction of RKN methods is

$$\frac{c_i^k}{k(k-1)} = \sum_{j=1}^{i-1} a_{ij} c_j^{k-2}, \qquad i = 1, 2, \ldots, s, \quad k = 2, 3, \ldots. \tag{3.1.4}$$

It is convenient to write non quadrature order conditions in terms of some functions introduced by Hairer [41]. These functions were later used by Dormand *et al.* in [20] for the construction of high order methods. These functions Q_{ij} and R_{kj} are defined by

$$Q_{ij} = \sum_{j=1}^{i-1} a_{ij} c_j^k - \frac{c_i^{k+2}}{(k+1)(k+2)}, \quad i = 1, 2, \ldots, s, \quad k = 1, 2 \ldots,$$

and

$$R_{kj} = \sum_{i=1}^{s} b'_i c_i^k a_{ij} - b'_j \frac{c_j^{k+2} - c_j(k+2) + (k+1)}{(k+1)(k+2)}, \quad j = 1, 2, \ldots, s, \quad k = 0, 1, 2 \ldots.$$

3.1.3 Leading truncation error coefficients

Once the order of a RKN method has been selected, the local truncation error (LTE) is an important measure of the accuracy of the method. The LTE for RKN methods is given by

$$\begin{aligned} \hat{t}_{n+1} = &\quad \hat{y}(x_n) + h_n \hat{\Phi}(x_n, \hat{y}_n, \hat{y'}_n, h_n) - \hat{y}(x_{n+1}), \\ \hat{t'}_{n+1} = &\quad \hat{y}'(x_n) + h_n \hat{\Phi}'(x_n, \hat{y}_n, \hat{y'}_n, h_n) - \hat{y}'(x_{n+1}). \end{aligned} \tag{3.1.5}$$

Expanding $y(x_{n+1})$ and $y'(x_{n+1})$ by Taylor series, the above equation takes the form

$$\hat{t}_{n+1} = h_n(\hat{\Phi}(x_n, \hat{y}_n, \hat{y'}_n, h_n) - \Delta(y(x_n), h_n)), \tag{3.1.6}$$

$$\hat{t'}_{n+1} = h_n(\hat{\Phi}(x_n, \hat{y}_n, \hat{y'}_n, h_n) - \Delta(y(x_n), h_n)). \tag{3.1.7}$$

Assuming f, $\hat{\Phi}$ and $\hat{\Phi}'$ have q^{th} order bounded continuous partial derivatives with respect to h, the truncation error terms may be written as [20, 26]

$$\hat{t}_{n+1} = \sum_{i=0}^{q-1} h_n^{i+1} \hat{\psi}_i(x_n, \hat{y}_n, \hat{y}'_n) + O(h_n^{q+1}), \tag{3.1.8}$$

$$\hat{t'}_{n+1} = \sum_{i=1}^{q-1} h_n^{i+1} \hat{\psi}'_i(x_n, \hat{y}_n, \hat{y}'_n) + O(h_n^{q+1}), \tag{3.1.9}$$

where

$$\hat{\psi}_i = \sum_{j=1}^{n_i} \hat{\tau}_j^{(i)} \hat{D}_j^i(x_n, \hat{y}_n(x), \hat{y}'_n(x)), \tag{3.1.10}$$

$$\hat{\psi'}_i = \sum_{j=1}^{n_i+1} \hat{\tau'}_j^{(i)} \hat{D}_j^{i+1}(x_n, \hat{y}_n(x), \hat{y}'_n(x)), \tag{3.1.11}$$

are known as error functions. The q^{th} order term in the above expressions is called the leading truncation error term.

The coefficients $\hat{\tau}_j^{(i)}$ and $\hat{\tau'}_j^{(j)}$ are the error coefficients for the RKN methods and are expressions involving a_{ij}, $\hat{b}_i$, c_i and a_{ij}, $\hat{b}'_i$, c_i respectively.

Definition 3.1.3 *A RKN method is of order q if $\hat{\psi}_i = \hat{\psi'}_i = 0$, $i = 0, \ldots, q-1$, and $\hat{\psi}_q$ and $\hat{\psi'}_q \neq 0$. The error functions $\hat{\psi}_q$ and $\hat{\psi'}_q$ are called the principal error functions.*

Definition 3.1.4 *The embedded RKN p-q pair is of order q and p, $(q > p)$ if the following holds*

$$\hat{\tau}_j^{(i+1)} = 0, \quad i = 1, \ldots, q-1, \quad j = 1, \ldots, n_i, \tag{3.1.12}$$

$$\hat{\tau'}_j^{(i+1)} = 0, \quad i = 1, \ldots, q-1, \quad j = 1, \ldots, n_{i+1},$$

$$\tau_j^{(i+1)} = 0, \quad i = 1, \ldots, p-1, \quad j = 1, \ldots, n_i, \tag{3.1.13}$$

$$\tau'^{(i+1)}_j = 0, \quad i = 1, \ldots, p-1, \quad j = 1, \ldots, n_{i+1}.$$

The above four system of equations are the order conditions for q^{th} and p^{th} order formulae. As discussed by Horn [51], the number of function evaluations, step-size estimates, truncation error analysis and stability characteristics play a significant role in determining the efficiency of a method.

3.2 Stability of RKN methods

Here we review some commonly used stability criteria for RKN methods.

3.2.1 Matrix stability criteria

Absolute stability analysis for RKN methods using a matrix stability criterion were first used by Chawla and Sharma [13, 14]. This analysis has been used by many researchers for example, Van der Houwen and Sommeijer [53], Sharp *et al.* [83], Van der Houwen *et al.* [55] and Paternoster [71]. In this analysis, the stability of RKN methods (3.1.1)-(3.1.2) is investigated by applying them to the test equation

$$y'' = -\lambda^2 y, \qquad \lambda \in \mathbb{R}. \tag{3.2.1}$$

This yields

$$y_{n+1} = y_n + hy'_n + zb^T K_n, \tag{3.2.2}$$

$$hy'_{n+1} = hy'_{n+1} + hzb'^T K_n, \tag{3.2.3}$$

where

$$K_n = N^{-1}(y_n + chy'_n), \tag{3.2.4}$$

with $N = (I - zA)$ and $z = -h^2\lambda^2$. The above set of equations may be written as

$$\begin{pmatrix} y_{n+1} \\ hy'_{n+1} \end{pmatrix} = R(z) \begin{pmatrix} y_n \\ hy'_n \end{pmatrix}, \tag{3.2.5}$$

where

$$R(z) = \begin{pmatrix} 1 + zb^T N^{-1} e & 1 + zb^T N^{-1} c \\ zb'^T N^{-1} e & 1 + zb'^T N^{-1} c \end{pmatrix},$$

with $A = (a_{ij})_{i,j=1}^{s}$, $e = (1, \ldots, 1)^T$, $b = (b_1, \ldots, b_s)^T$, $b' = (b'_1, \ldots, b'_s)^T$ and $c = (c_1, \ldots, c_s)^T$. The matrix $R(z)$ is called the stability matrix. Following Van der Houwen and Sommeijer [54], we introduce the functions $S(z)$ and $P(z)$

$$S(z) = \text{trace}(R(z)), \qquad P(z) = \det(R(z)).$$

It is easily shown that these $S(z)$ and $P(z)$ are algebraic polynomials. The interval of absolute stability is the values of z for which the spectral radius $\rho(R(z)) < 1$, and the interval of periodicity is the value of z for which $|R(z)| = 1$ and $S(z)^2 - 4P(z) < 0$ (see [55] for example).

3.2.2 Horn's stability criteria

Horn discussed the absolute stability for RKN methods in detail in [51] and recommended using the test equation

$$y'' = \lambda^2 y + g(x), \qquad \lambda \in \mathbb{C}, \tag{3.2.6}$$

where λ^2 is a constant. This test equation has since been used by Dormand *et al.* [20] and El-Mikkawy [26]. It is easily shown that when a RKN method is applied to the above test equation

$$\begin{aligned} y(x_0 + h) &= y_0\, P(h\lambda), \\ y'(x_0 + h) &= y'_0\, P'(h\lambda), \end{aligned}$$

where

$$P(h\lambda) = \sum_{k=0}^{2s} u_k (h\lambda)^k,$$

and

$$P'(h\lambda) = \sum_{k=0}^{2s-1} v_k (h\lambda)^k.$$

The expressions u_k and v_k are combinations of c_i, a_{ij}, b_i and b'_i and are detailed by Horn [51]. Since the error in y and y' may be magnified because of the polynomials P and P', the conditions $|P(h\lambda)| < 1$ and $|P'(h\lambda)| < 1$ should hold for a method to be stable for a particular value of $h\lambda$. The values of $h\lambda$ such that $|P(h\lambda)| = |P'(h\lambda)| = 1$ give the boundary of stability region for y and y' respectively.

The above completes our review of the order and stability definitions for RKN methods. We now review the derivation of the ERKN 8-10 pairs of El-Mikkawy [26], and the ERKN 10-12 pairs of Dormand *et al.* [21].

3.3 Solving the order conditions for 8-10 pairs

We have 235 equations to solve: 112 for $\hat{y}'$, 64 for $\hat{y}$, 37 for y' and 22 for y. El-Mikkawy [26] derived a family of 8-10 pair for $s = 13$ and imposed the following simplifying assumptions

in the form of Q_{ij} as

$$\left.\begin{aligned} Q_{i1} &= 0, \\ Q_{i2} &= 0, \end{aligned}\right\} \quad i = 3, \ldots, 13, \tag{3.3.1}$$

$$\left.\begin{aligned} Q_{i3} &= 0, \\ Q_{j4} &= 0, \end{aligned}\right\} \quad i = 5, \ldots, 13, \quad j = 6, \ldots, 13, \tag{3.3.2}$$

and

$$a_{i2} = 0, \qquad i = 5, \ldots, 13,$$

$$\hat{b}'_i, b'_i = 0, \qquad i = 2, \ldots, 5.$$

When the above stated simplifying assumptions are applied, the equations that remain are (equations (3.3.1) and (3.3.2)) along with the quadrature conditions

$$\hat{b}'_i c_i^k = \frac{1}{k+1}, \quad i = 1, \ldots, 13, \quad k = 1, \ldots, 10, \tag{3.3.3}$$

$$b'_i c_i^k = \frac{1}{k+1}, \quad i = 1, \ldots, 13, \quad k = 1, \ldots, 8, \tag{3.3.4}$$

and the equations

$$\left.\begin{aligned} \hat{b}'_i c_i^j Q_{i5} &= 0, \\ b'_i Q_{i5} &= 0, \end{aligned}\right\} \quad i = 6, \ldots, 13, \quad j = 0, 1, 2, \tag{3.3.5}$$

$$\hat{b}'_i c_i^j Q_{i6} = 0, \qquad i = 6, \ldots, 13, \quad j = 0, 1, \tag{3.3.6}$$

$$\hat{b}'_i Q_{i7} = 0, \qquad i = 6, \ldots, 13, \tag{3.3.7}$$

$$\left.\begin{aligned} \hat{b}'_i c_i^j a_{i3} &= 0, \\ b'_i a_{i3} &= 0, \end{aligned}\right\} \quad i = 6, \ldots, 13, \quad j = 0, 1, 2, \tag{3.3.8}$$

$$\left.\begin{aligned} \hat{b}'_i c_i^j a_{i4} &= 0, \\ b'_i a_{i4} &= 0, \end{aligned}\right\} \quad i = 6, \ldots, 13, \quad j = 0, 1, 2, \tag{3.3.9}$$

$$\hat{b}'_i c_i^j a_{i5} = 0, \qquad i = 6, \ldots, 13, \quad j = 0, 1. \tag{3.3.10}$$

The above system of non-linear algebraic equations is solved using the following steps.

i. We have nine non-zero exterior weights $\hat{b'}_i$ for the order 10 derivative formula. We use these non-zero weights to solve the quadrature conditions of orders one to nine for the derivative formula, see (3.3.3). The order ten quadrature condition from equation (3.3.3) is satisfied by constraining c_{12}.

ii. Equations (3.3.4) are then solved for the exterior weights b'_i, with b'_{13} as free parameters.

iii. The exterior weights of order 10 and 8 solution formulae are then found by using the transformation (3.1.3).

iv. Solve $Q_{31} = Q_{32} = 0$ from equation (3.3.1), for a_{32} and c_2.

v. Equations (3.3.1) and (3.1.4) are solved for $Q_{41},\ Q_{42}$, providing the values of $a_{41},\ a_{42}$ and a_{43}.

vi. We solve equations (3.3.1) for Q_{51} and Q_{52} yielding the values of a_{53} and a_{54}, putting back these values in Q_{53}, a quadratic equation in c_5 is obtained.

vii. A sixth row of interior weights (a_{ij} of coefficient matrix A) is obtained by solving $Q_{6i} = 0,\ i = 1, 2, 3$ and equation (3.1.4). Again plugging back these values into Q_{64}, a cubic equation in c_6 is achieved. We solve this cubic equation for c_3 and put it in the previously found quadratic equation, two more abscissae c_3 and c_4 are found.

viii. The seventh, eighth and ninth rows for interior weights are obtained using equations (3.3.1) and (3.3.2) with $i = 7, 8, 9$. This makes the interior weights $a_{83},\ a_{93}$ and a_{94} as free parameters.

ix. Now solving equations (3.3.8) and (3.3.9), it is easy to find the 3^{rd} and 4^{th} columns of interior weights.

x. It is convenient to use equations (3.3.5), (3.3.6) and (3.3.7) to obtain values for $Q_{i5},\ \ i = 10, \ldots, 13,\ \ Q_{i6},\ \ i = 12, 13$ and Q_{137} respectively.

xi. The 10^{th} and 11^{th} rows of the coefficient matrix can now easily be acquired using Q_{10i} and $Q_{11i},\ \ i = 1, \ldots, 5$, making a_{115} as free parameter.

xii. We solve equation (3.3.10) in a similar way as that discussed in the above step. The values of a_{125} and a_{135} are found.

xiii. Now straightforwardly the remaining interior weights for the 12^{th} and 13^{th} rows of coefficient matrix are earned after solving $Q_{12i},\ \ i = 1, \ldots, 6$, and $Q_{13i},\ \ i = 1, \ldots, 7$, respectively.

The derivation produces a family of 13-stage, 8-10 pairs with free parameters $c_i, i = 5, 6, \ldots, 11,\ a_{83}, a_{93}, a_{94}, a_{115}$ and $\hat{b'}_{13}$. We will find the suitable values for these free parameters in upcoming sections using search methods.

3.4 Solving the order conditions for 10-12 pairs

A family of 17-stage 10-12 pair was derived by Dormand *et al.* [21]. For a 10-12 pair, we need to solve a total of 732 equations: 357 and 199 equations for 12^{th} and 112 and 64 equations for 10^{th} order method for derivative and solution respectively. The order conditions are written in Q_{ij} and R_{0j} form. The simplifying assumptions used by Dormand *et al.* [21] are given by Baker *et al.* [4] and are

$$\left.\begin{aligned} Q_{i1} &= 0, \\ Q_{i2} &= 0, \end{aligned}\right\} \quad i = 3, \ldots, 17, \tag{3.4.1}$$

$$\left.\begin{aligned} Q_{i3} &= 0, \\ Q_{j4} &= 0, \end{aligned}\right\} \quad i = 5, \ldots, 17, \quad j = 6, \ldots, 17, \tag{3.4.2}$$

$$\left.\begin{aligned} R_{0j} &= 0, \\ \hat{R}_{0j} &= 0, \end{aligned}\right\} \quad j = 1, \ldots, 17, \tag{3.4.3}$$

$$\begin{aligned} a_{i2} &= 0, \qquad i = 5, \ldots, 17, \\ \hat{b}'_i, b'_i &= 0, \qquad i = 2, \ldots, 6. \end{aligned}$$

It is possible to eliminate all the order conditions except the following ones after applying the above simplifying assumptions. The remaining equations including the quadrature conditions are

$$\hat{b}'_i c_i^k = \frac{1}{k+1}, \quad i = 1, \ldots, 17, \ \ k = 1, \ldots, 12, \tag{3.4.4}$$

$$b'_i c_i^k = \frac{1}{k+1}, \quad i = 1, \ldots, 17, \ \ k = 1, \ldots, 10, \tag{3.4.5}$$

$$\hat{b}'_i c_i^j Q_{i5} = 0, \qquad i = 7, \ldots, 17, \ \ j = 0, 1, \ldots, 4, \tag{3.4.6}$$

$$b'_i c_i^j Q_{i5} = 0, \quad i = 7, \ldots, 17, \quad j = 0, 1, 2, \tag{3.4.7}$$

$$\hat{b}'_i c_i^j Q_{i6} = 0, \quad i = 7, \ldots, 17, \quad j = 0, 1, 2, 3, \tag{3.4.8}$$

$$b'_i c_i^j Q_{i6} = 0, \quad i = 7, \ldots, 17, \quad j = 0, 1, \tag{3.4.9}$$

$$\hat{b}'_i c_i^j Q_{i7} = 0, \quad i = 7, \ldots, 17, \quad j = 0, 1, 2, \tag{3.4.10}$$

$$\hat{b}'_i c_i^j Q_{i8} = 0, \quad i = 7, \ldots, 17, \quad j = 0, 1, \tag{3.4.11}$$

$$\hat{b}'_i c_i^j a_{i3} = 0, \quad i = 7, \ldots, 17, \quad j = 0, 1, \ldots, 4, \tag{3.4.12}$$

$$b'_i c_i^j a_{i3} = 0, \quad i = 7, \ldots, 17, \quad j = 0, 1, 2, \tag{3.4.13}$$

$$\hat{b}'_i c_i^j a_{i4} = 0, \quad i = 7, \ldots, 17, \quad j = 0, 1, \ldots, 4, \tag{3.4.14}$$

$$b'_i c_i^j a_{i4} = 0, \quad i = 7, \ldots, 17, \quad j = 0, 1, \ldots, 4, \tag{3.4.15}$$

$$\hat{b}'_i c_i^j a_{i5} = 0, \quad i = 7, \ldots, 17, \quad j = 0, 1, 2, 3, \tag{3.4.16}$$

$$b'_i c_i^j a_{i5} = 0, \quad i = 7, \ldots, 17, \quad j = 0, 1, \tag{3.4.17}$$

$$\hat{b}'_i c_i^j a_{i6} = 0, \quad i = 7, \ldots, 17, \quad j = 0, 1, 2. \tag{3.4.18}$$

The above equations can be solved as follows.

i. We get exterior weights for higher order formulae using quadrature conditions (3.4.4) up to order eleven in the form of the Vandermonde system, whilst $\hat{b}'_{17}$ is chosen as a free parameter, whereas the order twelve quadrature condition helps us to find the value for c_{15}.

ii. Similar is the case to obtain lower order weights using quadrature conditions (3.4.5) and assuming b'_{16} as free parameter and $b'_{17} = 0$.

iii. We adopt almost the same criteria as was done for 8-10 pair, the interior weights up to the ninth row are obtained using the equations (3.4.1) and (3.4.2) keeping a_{87}, a_{97} and a_{98} as free parameters.

iv. The coefficients a_{i3} and a_{i4}, $i = 10, \ldots, 15$, are found using equations (3.4.12), (3.4.13), (3.4.14) and (3.4.15).

v. On the next step, equations (3.4.6) and (3.4.7) yield Q_{i5}, $i = 10, \ldots, 15$. These are useful to find a_{10i} and a_{11i}, $i = 5, \ldots, 9$. Again managing a_{1110} as a free parameter.

vi. It is straightforward to get $a_{i5},\ i = 12, \ldots, 15$, by solving the equations (3.4.16) and (3.4.17).

vii. Equations (3.4.8) and (3.4.9) are solved for $Q_{i6},\ i = 12, \ldots, 15$. These along with equations (3.4.8) and (3.4.9) are helpful in attaining the rest of the 12^{th} and 13^{th} rows of the coefficient matrix. These are a_{12i} and $a_{13i},\ i = 6, \ldots, 13$, again adopting a_{1312} as a free parameter.

viii. Now we use equations (3.4.10) and (3.4.11) to obtain Q_{147}, Q_{157} and Q_{158}. But prior to calculating Q_{158}, it is handy to solve equation (3.4.18) in finding a_{146} and a_{156}.

ix. The interior weights $a_{14i},\ i = 7, \ldots, 13$, and $a_{15i},\ i = 7, \ldots, 14$, are attained using recently found Q_{i7} and Q_{i8} alongwith equations (3.4.10) and (3.4.11).

x. It is now effortless to achieve $a_{16i},\ i = 3, \ldots, 15$, and $a_{17i},\ i = 3, \ldots, 16$, using the first and second equations of (3.4.3) with $j = 3, \ldots, 15$ and $j = 3, \ldots, 16$, respectively.

The derivation produces a family of 17-stage, 10-12 pairs with free parameters $c_5, c_6, \ldots, c_{14}, a_{87}, a_{97}, a_{98},\ a_{1110},\ a_{1312}, \hat{b}'_{16}$ and b'_{17}.

3.5 Selecting a pair

The coefficients of a ERKN pair are obtained by solving order conditions and assigning the values for the free parameters. Generally, the technique used for the choice of free parameters is a grid search algorithm performed on a restricted set of free parameters as done by El-Mikkawy [26]. This results in a pair having several desirable properties. These properties are similar to those used for RK pairs by Prince and Dormand [72], Verner [93] and for the RKNG pair of Sharp and Fine [82]. The first and foremost property is the efficiency of a pair. For an efficient pair, the free parameters are chosen so that the pair has a small principal truncation error norm and reasonable absolute stability regions [26].

Before we find the suitable values for these free parameters in the 8-10 and 10-12 pairs, we briefly review simulated annealing, which is the method we used instead of a grid search to find suitable pairs. We chose simulated annealing because it does not require

derivative information about the objective function and can even do reasonably well on objective functions that are discontinuous (as is the case here).

3.6 Simulated annealing

The simulated annealing (SA) is a generic heuristic for global optimisation problems and was introduced independently by Kirkpatrik *et al.* [58] and Cerny [12].

In simulated annealing, the goal is to find a point in space at which a real valued objective function is minimised by trying random variations of the current solution in an analogous way. SA might not always find the optimal solution to a given problem. However, it almost finds a better solution than grid methods. If the objective function has steep maxima or minima, the probability of SA in finding them significantly decreases [91]. Specifically, it moves about randomly in the solution space looking for a solution that minimises the value of some objective function. Because it is generated randomly, a move may cause the objective function to increase, decrease or remain unchanged. A worse variation is accepted as the new solution with a probability that decreases as the computation proceeds. The probability of accepting a worse state that may increase the value of an objective function is given by the equation

$$P = e^{(-\Delta f/T)} > r, \tag{3.6.1}$$

where Δf is the change in evaluation of the objective function, T is the control parameter called the temperature and r is a random number between 0 and 1. The optimisation algorithm is described by Corana *et al.* [16].

3.7 Optimisation problem

Our goal is to reduce the leading error coefficient as much as possible, which will be used as an objective function for our optimisation problem. The magnitude of the coefficients, lower and upper bound for the abscissae and stability regions are taken as constraints.

This optimisation problem can be stated as

$$\begin{aligned}
&\text{minimise} && f : \max \|\hat{\tau}^{(q+1)}, \hat{\tau}'^{(q+1)}\|, \\
&\text{subject to} && e_1 : \text{abscissae are bounded, } 0 \le X_i \le 1, \\
& && e_2 : \max\{|a_{ij}|, |\hat{b}|, |\hat{b'}|, |b|, |b'|\} < b_{max} \text{ (bound)}, \\
& && e_3 : \text{(Horn's stability) } \hat{Z}_H > \hat{U}_H \text{(bound)}, \\
& && e_4 : \text{(Matrix stability) } \hat{Z}_M > \hat{U}_M \text{(bound)}, \\
& && e_5 : \frac{\max\|\hat{\tau}^{(q+2)}, \hat{\tau}'^{(q+2)}\|}{\max\|\hat{\tau}^{(q+1)}, \hat{\tau}'^{(q+1)}\|} < d_r(q+2) \text{ (bound)},
\end{aligned} \tag{3.7.1}$$

where $X_i,\ i = 1, \ldots, N$, are the free parameters. In some cases, we replace our objective function by $\max \|\hat{\tau}^{(q+2)}, \hat{\tau}'^{(q+2)}\|$.

We incorporated the constraints by using the simple penalty function of setting the objective function to a fixed value when a constraint was violated.

3.7.1 Searching ERKN 8-10 pairs

The family of 8-10 pairs has 12 free parameters: $c_i, i = 5, 6, \ldots, 11,\ a_{83}, a_{93}, a_{94}, a_{115}$ and $\hat{b'}_{13}$. El-Mikkawy [26] established that the performance of the pairs was insensitive to the value of the free parameters $a_{83}, a_{93}, a_{94}, a_{115}$ and $\hat{b'}_{13}$. This leaves the seven free parameters $c_i, i = 5, 6, \ldots, 11$.

Before applying simulated annealing, we performed two-dimensional grid searches over these seven free parameters. This was done to gain insight about what values of the free parameters would give near optimal pairs. This searching was done by fixing five free parameters and using a grid for the remaining two free parameters. We did the grid search for all pairs of free parameters against others. Below we discuss the results for some pairs of free parameters.

Figures 3.1 and 3.2 show $\hat{Z}_H$ and $\hat{Z}_M$ respectively for different pairs of free parameters. Figure 3.1 contains two plots. The upper plot gives $\hat{Z}_H$ as a function of c_5 and c_7, and the lower plot gives $\hat{Z}_H$ as a function of c_5 and c_9. We observe from both plots that $\hat{Z}_H$ has a maximum for $c_5 < 0.4$. Figure 3.2 shows the region for $\hat{Z}_M$ plotted as a function

of c_5 and c_6 (top plot), and c_5 and c_{11} (bottom plot). We observe that the results are not as clear cut as in Figure 3.1. In the top plot, the maximum for $\hat{Z}_M$ occurs for c_5 in [0.4,0.5], whereas in the bottom plot the maximum occurs for $c_5 < 0.4$.

As noted above, we did not see a pattern for $\hat{Z}_M$. Hence, we investigated the dependence of $\hat{Z}_H$ and $\tau^{(11)}$ on pairs of free parameters. In Table 3.1, each of free parameters $c_i,\ i = 5, 6, \ldots, 11$, is plotted against others for maximum value of $\hat{Z}_H$ and minimum value of $\tau^{(11)}$. Our conclusions for the above seven free parameters are summarised in the following table.

	$\max(\hat{Z}_H)$	$\min(\tau^{(11)})$
c_5	$c_5 < 0.5$	$c_5 < 0.5$
c_6	$c_6 < 0.5$	$c_6 < 0.2$
c_7	$c_7 < 0.45$	$c_7 < 0.23$
c_8	$c_8 < 0.5$	$c_8 < 0.2$
c_9	$c_9 \in [0.45, 0.75]$	$c_9 \in [0.45, 0.55]$
c_{10}	$c_{10} \in [0.25, 1]$	$c_{10} \in [0.65, 0.7]$
c_{11}	$c_{11} \in [0, 1]$	$c_{11} \in [0.8, 1]$

Table 3.1: A summary of our conclusions about the ranges of the free parameters for the ERKN 8-10 pair that will lead to near optimal ERKN pairs.

Having completed the investigation using pairs of free parameters, we then used simulated annealing to find local minima to the optimisation problem (3.7.1). The optimisations were done for different values of the bounds b_{max}, $\hat{U}_Z$ and $\hat{U}_M$. We initially used the bounds employed by El-Mikkawy [26], and then tried smaller and larger bounds. The optimisations took a considerable amount of CPU time and led to 73 ERKN pairs. We tested all 73 ERKN pairs on some test problems and used the results of these tests to reduce the 73 ERKN pairs to 10.

Table 3.2 lists some of the properties of the ten new 8-10 pairs. The ERKN pairs are denoted by 8-10-p1 and so-on. The ERKN pair 8-10-edp represents the pair selected by El-Mikkawy in [26]. The ERKN pairs 8-10-p1 to 8-10-p8 were obtained using $\hat{\tau}^{(11)}$ as the objective function, subject to all five constraints in problem (3.7.1). The pairs 8-10-p9 and 8-10-p10 are earned using $\hat{\tau}^{(12)}$ as objective function solely. Table 3.2 shows that all ten ERKN pairs have a leading error coefficient $\hat{\tau}^{(11)}$ that is noticeably smaller than for the 8-10-edp pair of El-Mikkawy [26]. We also see that $\hat{Z}_M$ for some pairs is a lot smaller than for the 8-10-edp pair. We come back to this point later when discussing

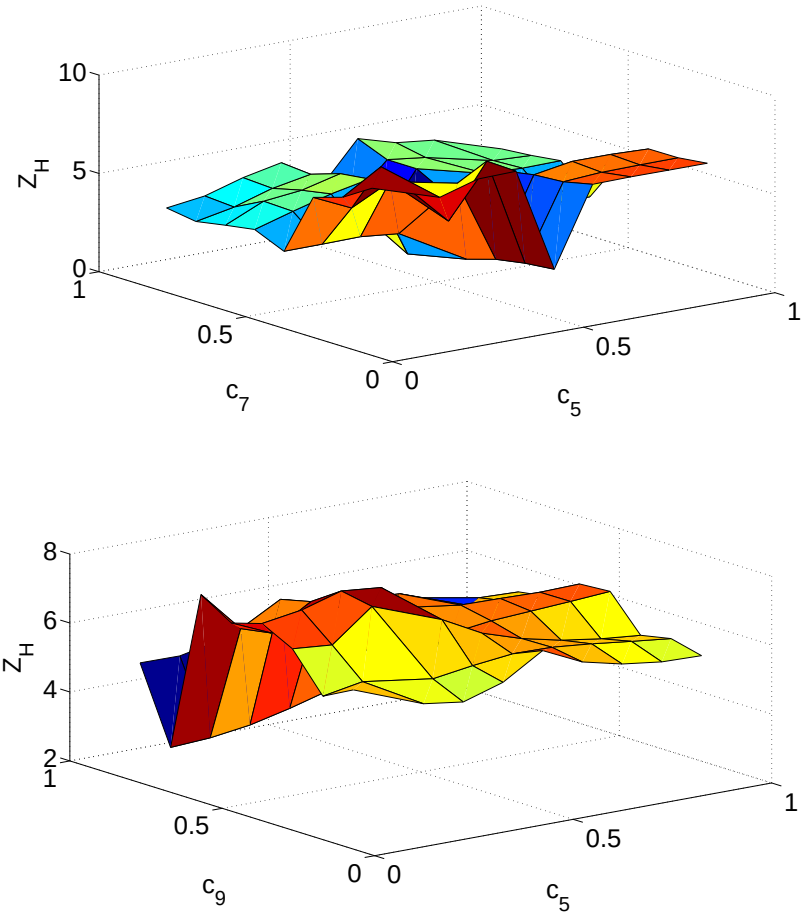

Figure 3.1: The plot of the stability interval $\hat{Z}_H$ for the 8-10 pair as a function of two free parameters. (Top) - The free parameters are c_5 and c_7. (Bottom) - The free parameters are c_5 and c_9.

Method	$\|\tau^{(11)}\|_\infty$	$\|\tau^{(12)}\|_\infty$	$\hat{Z}_H$	Z_H	$\hat{Z}_M$	Z_M	max(Mag.)
8-10-edp	0.1886	4.31	7.56	6.9	6.75	0.05	24.4
8-10-p1	0.0017	0.47	7.50	8.2	5.60	0.05	4.3
8-10-p2	0.0051	0.28	7.52	7.5	0.15	0.05	3.7
8-10-p3	0.0050	0.32	7.60	7.1	0.15	0.05	5.3
8-10-p4	0.0016	0.46	7.50	7.9	5.60	0.05	5.0
8-10-p5	0.0050	0.31	7.30	6.6	0.21	0.05	3.7
8-10-p6	0.0063	0.37	7.60	7.0	0.15	0.05	4.7
8-10-p7	0.0067	0.65	7.11	6.0	0.15	0.05	10.0
8-10-p8	0.0094	1.13	6.82	7.0	0.15	0.05	4.0
8-10-p9	0.0230	0.14	7.61	7.0	0.22	0.05	2.8
8-10-p10	0.0193	0.12	7.50	7.1	0.21	0.05	2.7

Table 3.2: Some properties of the ten 8-10 pairs that remained after our preliminary numerical testing of the 73 pairs.

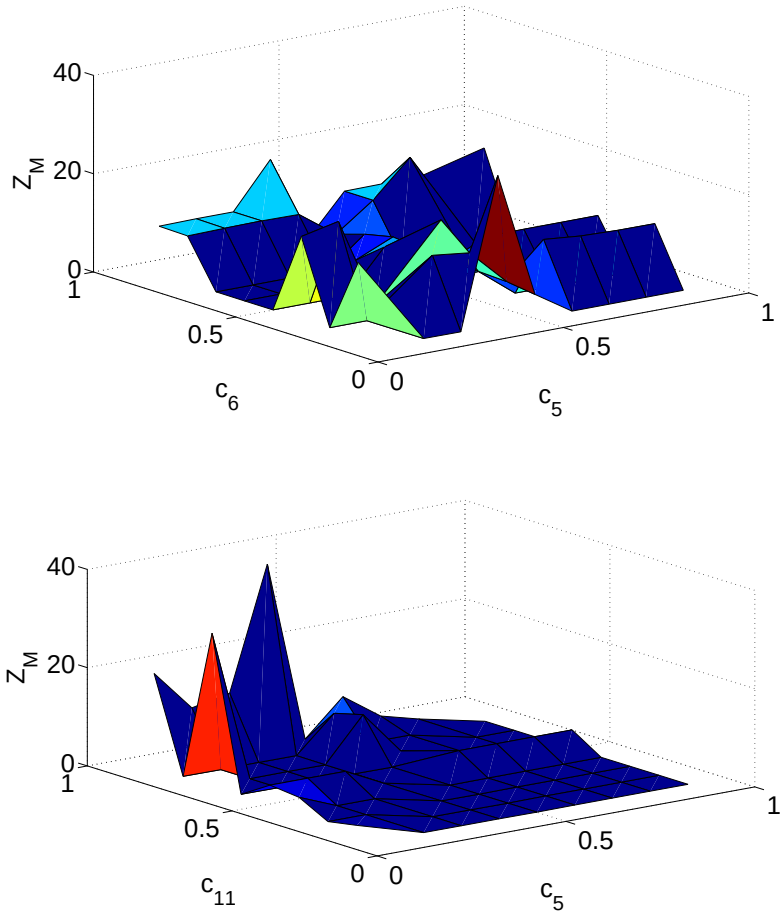

Figure 3.2: The plot of the stability interval $\hat{Z}_M$ for the 8-10 pair as a function of two free parameters. (Top) - The free parameters are c_5 and c_6. (Bottom) - The free parameters are c_5 and c_{11}.

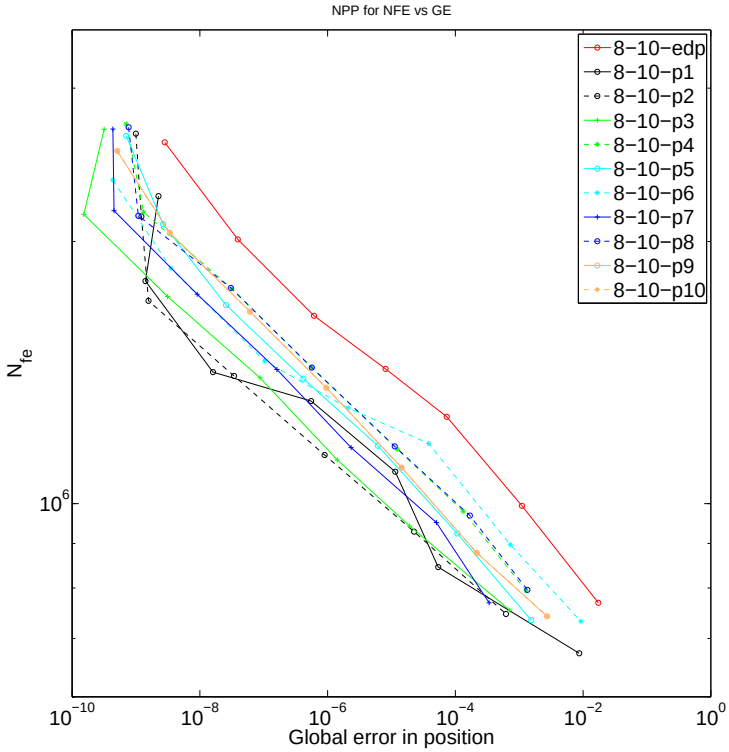

Figure 3.3: The efficiency plots for the eleven ERKN 8-10 pairs applied to the Nine Planets Problem. The interval of integration is one thousand years and the tolerances vary from 10^{-14} to 10^{-8}. The efficient plots were calculated from the norm of the maximum global error and the number of function evaluations.

the numerical testing.

The numerical tests are done in two parts. In the first part, we plotted the efficiency curves for some of the gravitational problems at a range of tolerances for a short interval of time (to avoid an excess amount of CPU time). In the second part, we tested the more efficient pairs from the first part on longer intervals of integration. The results for the second part are discussed later in the chapter. Here we discuss the results for the first part.

The test problems and intervals for the first part were the Nine Planet Problem over 1000 years, the HRC Problem over 10,000 days and the Jovian Problem over one million years. The efficiency curves for these pairs are plotted in Figures 3.3, 3.4 and 3.5. The red line represents the 8-10-edp pair of El-Mikkawy [26], while other colours correspond to ERKN new pairs. We employed local error tolerances in the range 10^{-14} to 10^{-8} and used the number of function evaluations as the measure of work. This is an acceptable way of measuring effort since all the pairs have the same number of stages, our results are

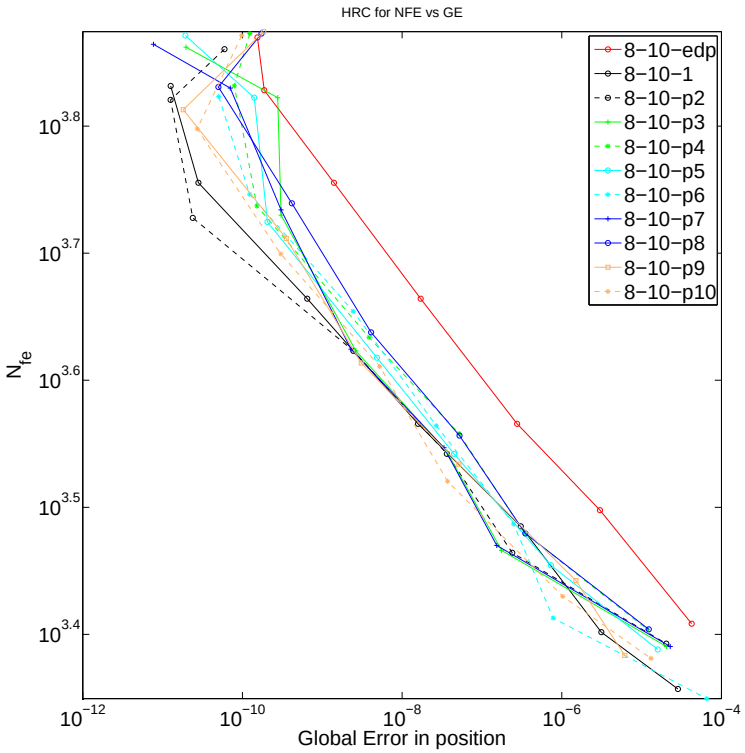

Figure 3.4: The efficiency plots for the eleven ERKN 8-10 pairs applied to the HRC Problem. The interval of integration is ten thousand days and the tolerances vary from 10^{-14} to 10^{-8}. The efficient plots were calculated from the norm of the maximum global error and the number of function evaluations.

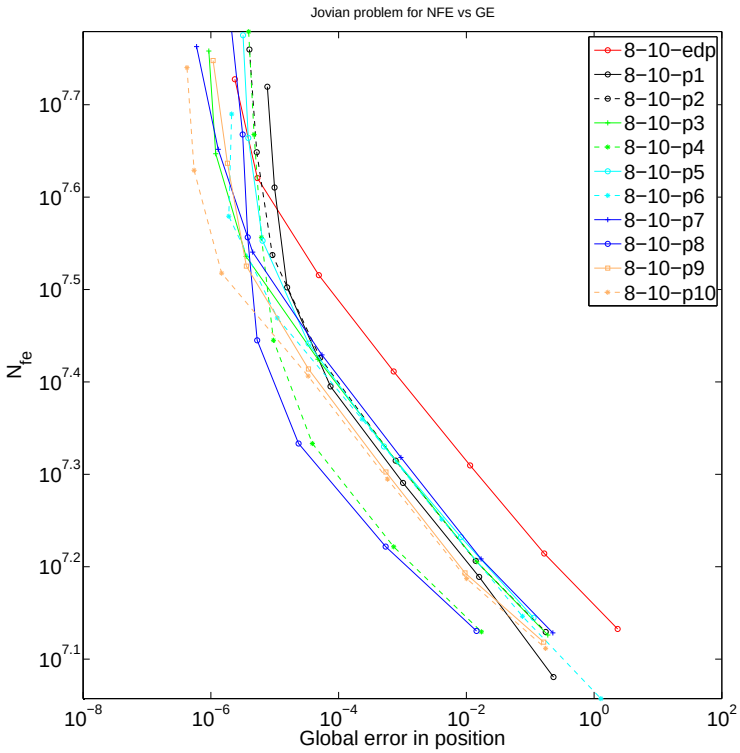

Figure 3.5: The efficiency plots for the eleven ERKN 8-10 pairs applied to the Jovian Problem. The interval of integration is one million years and the tolerances vary from 10^{-14} to 10^{-8}. The efficient plots were calculated from the norm of the maximum global error and the number of function evaluations.

thus essentially machine independent.

Figure 3.3 gives the efficiency curves for the Nine Planets Problem. We observe that all new pairs are more efficient than the 8-10-edp pair. In addition, we observe that the most accurate solution produced by the new pairs is more accurate than that produced by the 8-10-edp pair. The vertical segments in the curves for severe tolerances are due to round-off error. The conclusions are similar for the HRC Problem, see Figure 3.4. For the Jovian Problem, see Figure 3.5, we observe that at severe tolerances some of the new pairs are less efficient than the 8-10-edp pair. It is clear from the numerical tests and from the Table 3.2 that a larger or smaller stability interval for $\hat{Z}_M$ has no effect on the efficiency of the pairs.

We found the increase in efficiency of the new pairs relative to the 8-10-edp pair was less than predicted by the size of $\tau^{(11)}$ and $\tau'^{(11)}$. Nevertheless, there was qualitative agreement with the size of $\tau^{(11)}$ and $\tau'^{(11)}$.

3.7.2 Searching ERKN 10-12 pairs

In the case of the ERKN 10-12 pair, we have 17 free parameters: $c_5, c_6, \ldots, c_{14}$, a_{87}, a_{97}, a_{98}, a_{1110}, a_{1312}, $\hat{b}'_{16}$ and b'_{17}. By experiment we showed that $\hat{b}'_{16}$ and b'_{17} do not have much effect on the minimisation of the leading error coefficient $\tau^{(13)}$. So we use the same values of $\hat{b}'_{16}$ and b'_{17} as in ERKN 10-12 pair of Dormand *et al.* [21], namely 0.02 and 0.025 respectively.

We analyzed the behaviour of the coefficients for the ERKN 10-12 pair in the same way as was done for the 8-10 pair in the previous section. Figures 3.6 and 3.7 show the plots of $\hat{Z}_H$ and $\hat{Z}_M$ respectively. In the upper plot of Figure 3.6, $\hat{Z}_H$ is a function of c_5 and c_6 and function of c_5 and c_{13} in the lower plot. For both plots, the maximum of stability interval is obtained for value $c_5 \simeq 0.5$ and in [0.6,0.9] for c_5. We observe from the experiments that $\hat{Z}_M$ has no particular pattern for the free parameters, which is the same as for the 8-10 pair. This can be seen in Figure 3.7, the maximum stability interval occurs for c_5 near to 0.1 in the top plot, and close to 0.4 in the bottom plot.

Dependance of $\hat{Z}_H$ and $\tau^{(13)}$ on pairs of free parameters is summarised in the fol-

lowing table.

	$\max(\hat{Z}_H)$	$\min(\tau^{(13)})$
c_5	$c_5 \in [0.2, 0.4] \cup [0.6, 0.9]$	$c_5 \in [0, 1]$
c_6	$c_6 < 0.2$	$c_6 < 0.8$
c_7	$c_7 < 0.06$	$c_7 < 0.07$
c_8	$c_8 < 0.2$	$c_8 < 0.22$
c_9	$c_9 \in [0.22, 0.6]$	$c_9 \in [0.2, 0.35]$
c_{10}	$c_{10} \in [0.26, 0.41]$	$c_{10} \in [0.25, 0.35]$
c_{11}	$c_{11} \in [0.33, 0.7]$	$c_{11} \in [0.45, 0.52]$
c_{12}	$c_{12} \in [0.4, 0.6]$	$c_{12} \in [0.46, 0.56]$
c_{13}	$c_{13} \in [0.65, 1]$	$c_{13} \in [0.5, 0.76]$
c_{14}	$c_{14} \in [0, 1]$	$c_{14} \in [0.8, 0.9]$

Table 3.3: A summary of our conclusions about the ranges of the free parameters for the ERKN 10-12 pair that will lead to near optimal pairs.

Optimising the objective function $\tau^{(13)}$ in the same way as detailed in the previous section, 52 new ERKN pairs are obtained. Some of the properties of the seven most efficient new ERKN pairs are given in Table 3.4. The pair 10-12-dep represents the pair selected by Dormand *et al.* [21]. All other listed pairs are obtained using $\tau^{(13)}$ as the objective function with different values for d_r. Table 3.4 shows that all seven ERKN pairs have a leading error coefficient $\tau^{(13)}$ that is noticeably smaller than for the 10-12-dep pair of Dormand *et al.* [21]. We also see that $\hat{Z}_M$ for some pairs is a lot smaller than for the 10-12-dep pair as was previously observed for 8-10 pairs.

We tested these new 10-12 pairs on the same problems as used for the 8-10 pairs. Figures 3.8, 3.9 and 3.10 show the efficiency of new searched 10-12 pairs for the Nine Planets, HRC and Jovian Problems respectively for the same time interval as was done for 8-10 pairs. The gain in efficiency compared with the 10-12-dep pair is less than the gain for 8-10 pairs. The red line represents the 10-12-dep pair of Dormand *et al.* [21], while other colours correspond to ERKN new pairs.

Our numerical experiments show that only 10-12-p5 is more efficient for the Nine Planets Problem, when compared with the 10-12-dep pair on severe tolerances, see Figure 3.8. But most of the pairs are giving better efficiency on lax tolerances. Figure 3.9 gives the efficiency curves for the HRC Problem. We observe that all new pairs use fewer function evaluations than 10-12-dep pair at small tolerances. For the Jovian Problem,

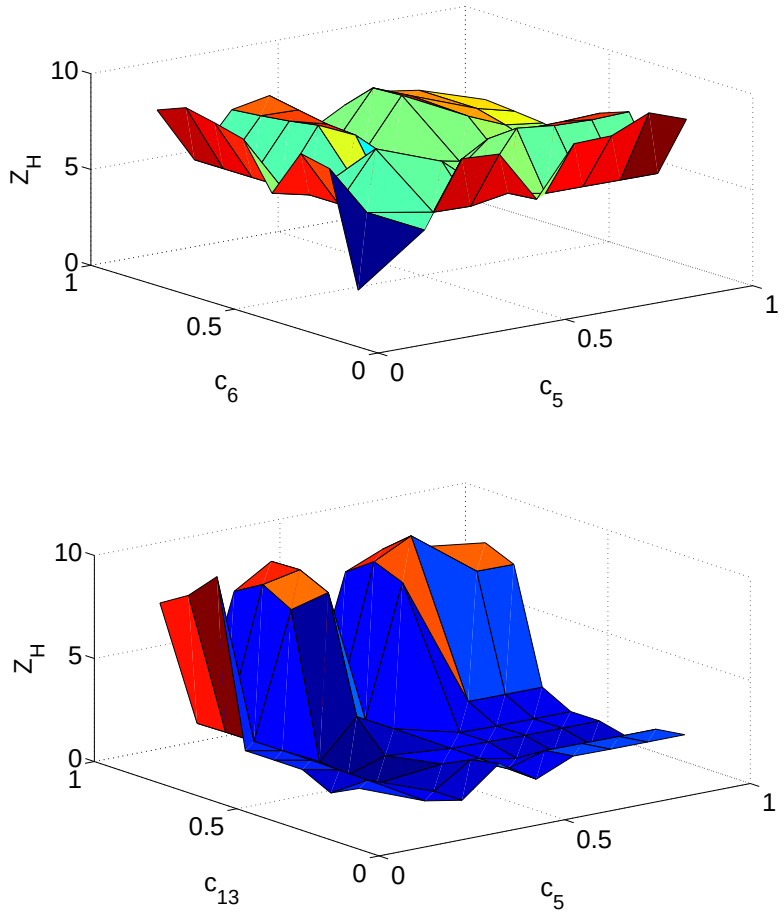

Figure 3.6: The plot of the stability interval $\hat{Z}_H$ for the 10-12 pair as a function of two free parameters. (Top) - The free parameters are c_5 and c_6. (Bottom) - The free parameters are c_5 and c_{13}.

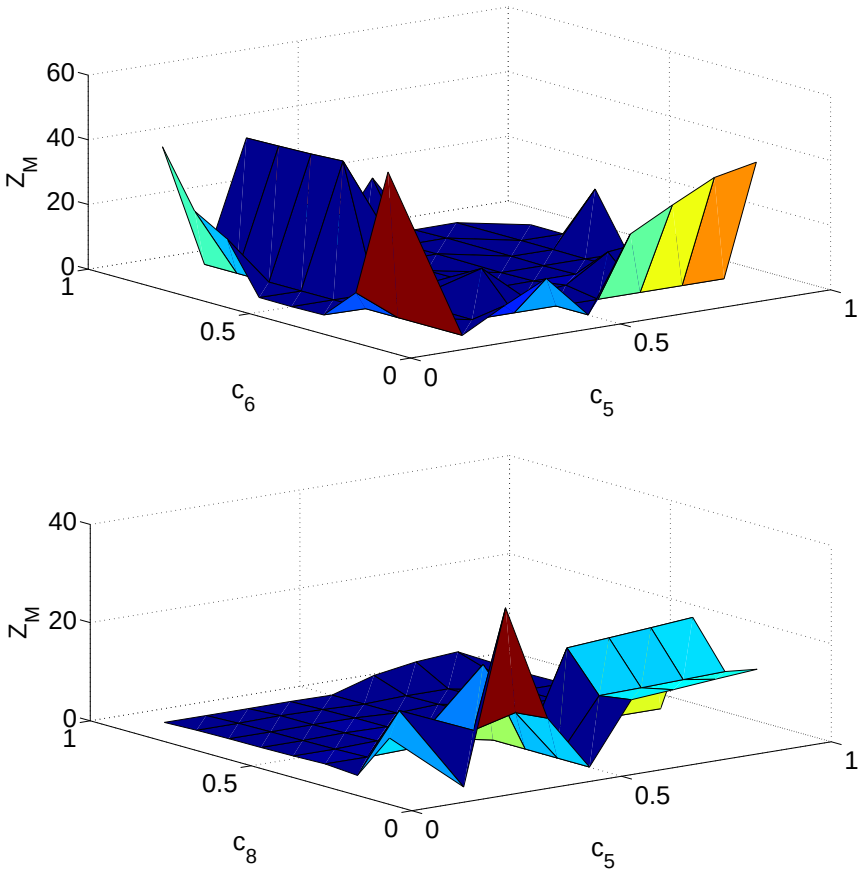

Figure 3.7: The plot of the stability interval $\hat{Z}_M$ for the 10-12 pair as a function of two free parameters. (Top) - The free parameters are c_5 and c_6. (Bottom) - The free parameters are c_5 and c_{13}.

Method	$\|\tau^{(13)}\|_\infty$	$\|\tau^{(14)}\|_\infty$	$\hat{Z}_H$	Z_H	$\hat{Z}_M$	Z_M	Mag.
10-12-dep	0.2600	7.94	10.54	10.2	0.25	35.25	48.95
10-12-p1	0.0018	3.97	10.11	10.1	0.65	0.15	39.85
10-12-p2	0.0060	3.97	10.93	10.2	0.45	0.15	48.96
10-12-p3	0.0041	3.97	9.13	8.7	0.41	0.11	48.96
10-12-p4	0.0037	3.94	7.80	7.0	0.20	12.10	5.54
10-12-p5	0.0023	3.97	10.53	10.2	8.35	8.25	48.95
10-12-p6	0.0078	3.26	6.54	5.5	0.21	0.13	35.89
10-12-p7	0.0052	3.63	7.06	7.0	0.35	2.05	13.32

Table 3.4: Some properties of the seven 10-12 pairs that remained after our preliminary numerical testing of the 52 pairs.

all new pairs are more efficient than 10-12-dep except 10-12-p5 at small tolerances, see Figure 3.10. The pair 10-12-p3 gives the least global error at tolerances 10^{-14}, 10^{-13} and 10^{-12}. The vertical segments in the curves for severe tolerances 10^{-14} to 10^{-12} are because of round-off error.

3.8 Numerical tests for long-term integration

In this section, we analyse the error growth for the position of planets and conserved quantities for long-term integration of the Solar System. We integrate the Jovian Problem, the Nine Planets Problem, the HRC Problem and the Saturnian Satellites problem for 100 million years, one million years, 10 thousand days and 27 thousand Earth years respectively.

To keep the CPU time requirement reasonable, we restrict ourselves to test the three most efficient pairs from the previous testing. This is done for the 8-10 and 10-12 pairs. We keep the number of function evaluations same and use the tolerances of 10^{-13} and 10^{-10}.

We take the pairs 8-10-p1, 8-10-p6 and 8-10-p10, for convenience we denote them as 8-10-p1, 8-10-p2 and 8-10-p3 respectively. Among the 10-12 pairs, we choose 10-12-p3, 10-12-p4 and 10-12-p6 and denote them as 10-12-p1, 10-12-p2 and 10-12-p3 respectively. We explore the behaviour of error growth for the pairs, and the efficiency at lax and severe tolerances. Some of the plots have high frequency fluctuations; we smooth that data using

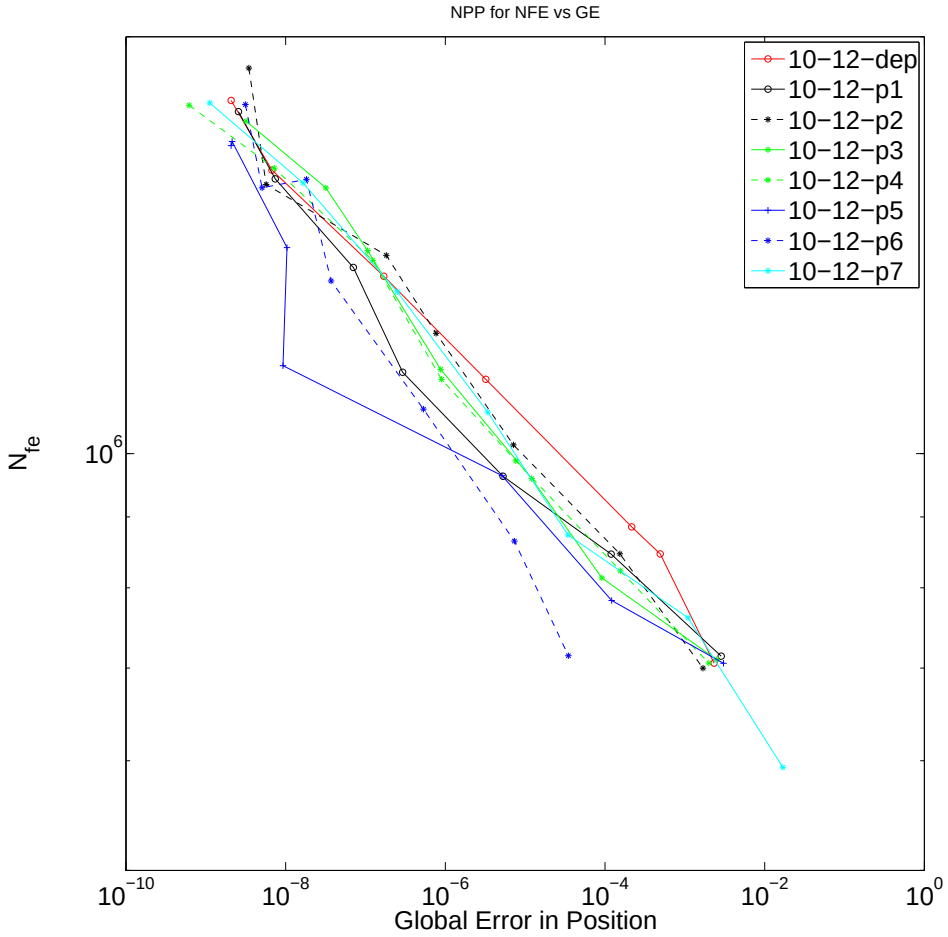

Figure 3.8: The efficiency plots for the eight ERKN 10-12 pairs applied to the Nine Planets Problem. The interval of integration is one thousand years and the tolerances vary from 10^{-14} to 10^{-8}. The efficient plots were calculated from the norm of the maximum global error and the number of function evaluations.

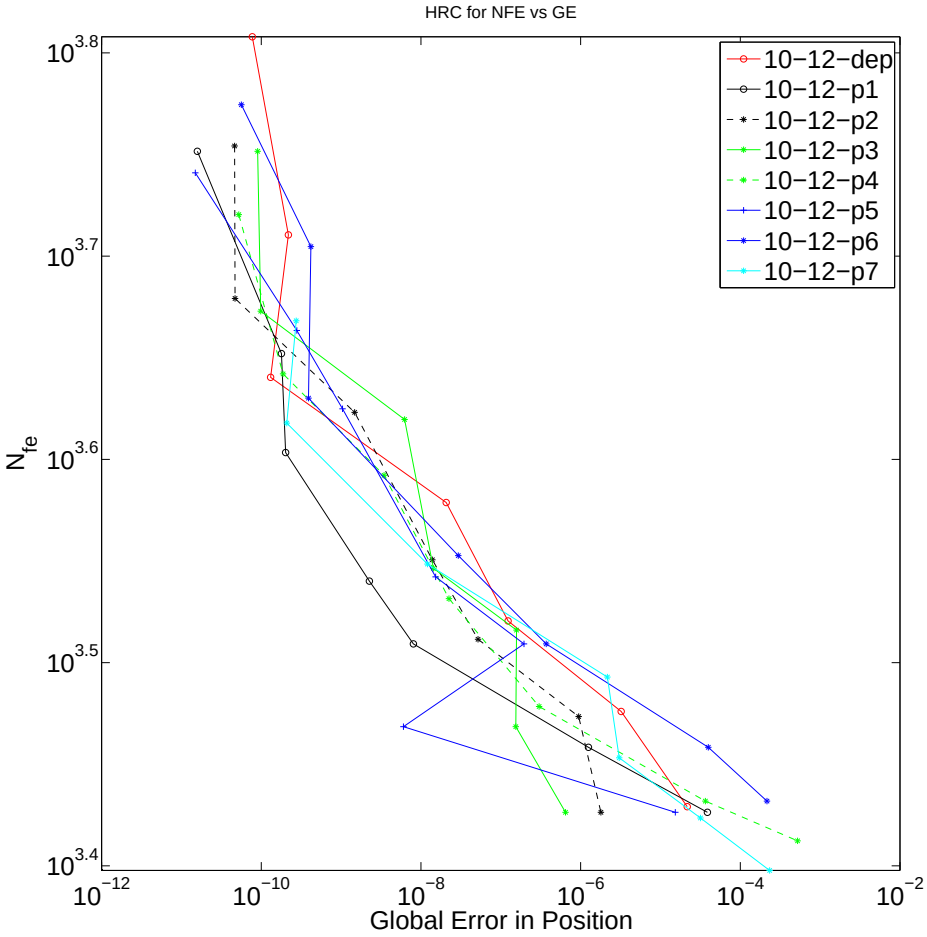

Figure 3.9: The efficiency plots for the seven ERKN 10-12 pairs applied to the HRC Problem. The interval of integration is one thousand days and the tolerances vary from 10^{-14} to 10^{-8}. The efficient plots were calculated from the norm of the maximum global error and the number of function evaluations.

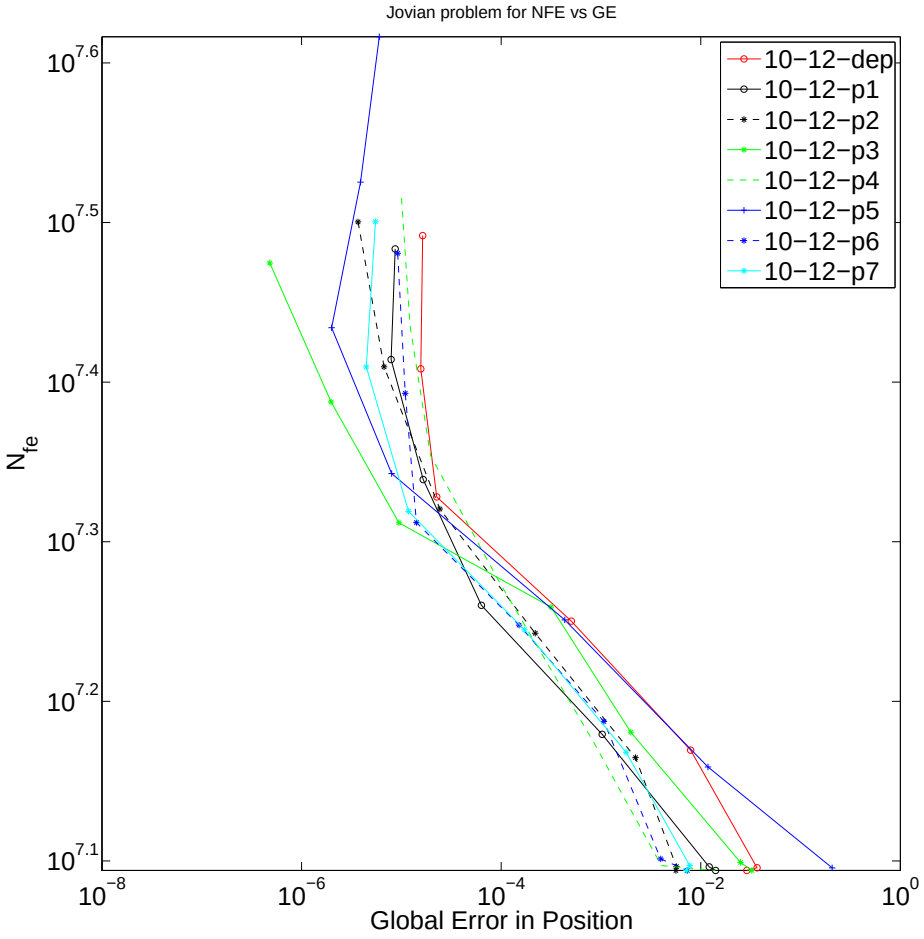

Figure 3.10: The efficiency plots for the seven ERKN 10-12 pairs applied to the Jovian Problem. The interval of integration is one million years and the tolerances vary from 10^{-14} to 10^{-8}. The efficient plots were calculated from the norm of the maximum global error and the number of function evaluations.

the 'filter' command in Matlab. The window size employed is 5% of the original data used. The percentage efficiency of the pairs is measured relative to the 8-10-edp and 10-12-dep pairs using the following formula

$$\left(\left(\frac{\text{error in ref. pair}}{\text{error in new pair}}\right)^{1/(p+1)} - 1\right) \times 100.$$

In the above formula, 'ref. pair' means the 8-10 pair of El-Mikkawy [26] and 10-12 pair of Dormand *et al.* [21]. The error is calculated at evenly spaced points with in the interval of integration. We found the reference solution by performing accurate simulations in quadruple precision with tolerances of 10^{-20}. The above formula is calculated at start, middle and end point of each integration.

Figure 3.11 shows the error growth in the position, the Hamiltonian and the angular momentum for the Jovian Problem using 8-10 pairs. The integration is done using a tolerance of 10^{-13} for the 8-10-edp pair, while for the other pairs, the tolerance is adjusted so they use the same number of evaluations as the 8-10-edp pair. It is evident from the figure that all new pairs give better accuracy than the 8-10-edp pair. The pairs 8-10-p1 and 8-10-p2 are approximately 30% more efficient, while 8-10-p3 is 56% more efficient than the 8-10-edp pair. The efficiency in the Hamiltonian and the angular momentum is similar to that for the global error. Similar error growth and efficiency were found, when comparing the pairs using the base tolerance 10^{-10}, except the 8-10-p1 pair which gives 99% more efficiency than the 8-10-edp pair. The error growth of angular momentum for 8-10-p3 in the bottom plot of Figure 3.11 is because round-off error dominates, while using the tolerance of 10^{-10} linear error growth is seen.

The Jovian Problem is also integrated using 10-12 pairs on both tolerances i.e. 10^{-10} and 10^{-13}. The top plot in Figure 3.12 shows that the 10-12-p1 is 18% more efficient than the 10-12-dep pair when computing the error in position. The pair 10-12-p2 gives almost the similar accuracy as for the 10-12-dep pair. The gain in efficiency when the error in Hamiltonian and angular momentum are used is approximately similar for these three pairs. On the other hand, 10-12-p3 proved to be 10% more expensive than the 10-12-dep pair. For the tolerance 10^{-10}, the pairs 10-12-p1, 10-12-p2 become approximately 14% and 25% more efficient respectively. The 10-12-p3 pair gives almost the similar accuracy, exhibiting only 3% efficiency which is not much worthy. This means that at bigger tolerance new pairs are behaving better than at smaller tolerance for the Jovian Problem. (See Tables 3.7 and 3.8.)

Pair	$b(\ \mathcal{E}_{\rm ge}\ )$	$b(\ \mathcal{E}_{\rm H}\ )$	$b(\ \mathcal{E}_{\rm L}\ )$
8-10-dep	2.17	0.97	0.97
8-10-p1	2.18	0.97	0.89
8-10-p2	2.18	0.97	0.94
8-10-p3	2.20	0.99	0.78
10-12-dep	2.07	0.92	0.89
10-12-p1	2.05	0.91	0.92
10-12-p2	2.07	0.91	0.90
10-12-p3	2.07	0.85	0.83

Table 3.5: The exponent b of the power law for global error, relative error in the Hamiltonian and angular momentum for the Jovian Problem with a local error tolerance 10^{-13}.

We also did a linear least square fit for the power law ax^b and found the values of b close to the theory as detailed in Table 3.5.

We also simulate the Nine Planets Problem in a similar manner to that for the Jovian Problem, keeping the same number of function evaluations on two different tolerances. Figure 3.13 presents the error growth in the position, the Hamiltonian and the angular momentum for 8-10 pairs applied to the Nine Planets Problem using a tolerance 10^{-13}. We observe that 8-10-p1, 8-10-p2 and 8-10-p3 are respectively 30%, 62% and 20% more efficient than the 8-10-edp pair when computing the positional error. The efficiency for the error in the Hamiltonian is 27% and 20% more than 8-10-edp for 8-10p1 and 8-10-p2 respectively, while the pair 8-10-p3 is 52% more efficient than for 8-10-edp. The growth behaviour seems to be affected by round-off error. In the bottom plot of Figure 3.13 the growth in the angular momentum is measured for these pairs, which is almost the same for all pairs because of the dominance of the round-off error. The experiment with keeping the tolerance 10^{-10} shows that 8-10-p1, 8-10-p2 and 8-10-p3 become 27%, 50% and 37% more efficient than the 8-10-edp pair. The pairs 8-10-p2 and 8-10-p3 gain 41% and 18% more efficiency for the Hamiltonian and a similar amount for the angular momentum (see Table 3.8).

Figures 3.14 and 3.15 show the error growth for 10-12 pairs using the tolerances 10^{-13} and 10^{-10} respectively. The pair 10-12-p3 proved to be approximately 10 to 17% more expensive than the 10-12-dep pair at both tolerances. The pairs 10-12-p1 and 10-12-p2 become more efficient than the 10-12-dep at tolerance 10^{-10} and gain efficiency up to 26% while computing global error. Similar was the case for the efficiency for the error in Hamiltonian. But for angular momentum, 10-12-p1 and 10-12-p2 only give 6% efficiency,

Pair	$b(\ \mathcal{E}_{\rm ge}\ )$	$b(\ \mathcal{E}_{\rm H}\ )$	$b(\ \mathcal{E}_{\rm L}\ )$
8-10-dep	2.25	1.00	0.72
8-10-p1	2.23	0.97	0.73
8-10-p2	2.23	0.95	0.72
8-10-p3	2.20	0.74	0.73
10-12-dep	2.24	0.81	0.71
10-12-p1	2.24	0.94	0.72
10-12-p2	2.19	0.72	0.70
10-12-p3	2.24	0.99	0.73

Table 3.6: The exponent b of the power law for global error, relative error in the Hamiltonian and angular momentum for the Nine Planets Problem with a local error tolerance 10^{-13}.

this may be because of round-off error. The global error in the top plot of Figure 3.15 has the quadratic error growth of up to 10^5 years, while it starts oscillating afterwards for all pairs.

The error growth for H and L in Figure 3.14 shows oscillations. We repeated the integrations with tolerance increased from 10^{-13} to 10^{-12} and found the oscillations disappeared, indicating that round-off error was causing the oscillations. This is because as the base tolerance is increased, the error growth becomes linear. (See the middle and bottom plot of Figure 3.15 .)

Table 3.6 shows the exponent of power law, which is slightly bigger than expected for $\mathcal{E}_{\rm ge}$. The exponent b for $\mathcal{E}_{\rm H}$ are in reasonable agreement to the expected values except for 8-10-p3 and 10-12-p2. The values of b for $\mathcal{E}_{\rm L}$ are smaller than expected, but are consistent across the six pairs, suggesting an underlying cause such as round-off error.

Figures 3.16 and 3.17 show the error growth in position for the HRC Problem using the 8-10 and 10-12 pairs at tolerances 10^{-13} and 10^{-10} respectively. In both plots, the global error up to 2200 days is approximately the same for all pairs and suddenly increases afterwards. This increase is due to close approaches of the comet to jupiter as previously discussed in Chapter 1. At the end point of integration, 8-10-p1, 8-10-p2 and 8-10-p3 give 39%, 57% and 25% increase in efficiency respectively using the tolerance 10^{-13}.

The pairs 10-12-p1 and 10-12-p2 give 11% and 13% less error, while 10-12-p3 proves to be 6% more expensive in error growth. The experiment shows that for the HRC Problem, 8-10-p1 and 8-10-p2 behave more efficiently for tolerance 10^{-13} than using 10^{-10}.

		$\mathcal{E}_{\mathrm{ge}}$		$\mathcal{E}_{\mathrm{H}}$		$\mathcal{E}_{\mathrm{L}}$	
		8-10	10-12	8-10	10-12	8-10	10-12
JOV	p1	30%	18%	31%	24%	28%	22%
	p2	30%	2%	26%	2%	22%	2%
	p3	56%	−10%	54%	−10%	62%	−11%
NPP	p1	30%	−1%	27%	−17%	−1%	−1%
	p2	62%	1%	20%	−1%	1%	1%
	p3	20%	−17%	52%	−27%	1%	1%
HRC	p1	39%	2%	–	–	–	–
	p2	57%	3%	–	–	–	–
	p3	25%	−4%	–	–	–	–
SS	p1	17%	15%	–	–	–	–
	p2	−9%	21%	–	–	–	–
	p3	−3%	24%	–	–	–	–

Table 3.7: Percentage efficiency calculated from the global error in position, the relative error in Hamiltonian and the angular momentum, all for a local error tolerance 10^{-13}.

		$\mathcal{E}_{\mathrm{ge}}$		$\mathcal{E}_{\mathrm{H}}$		$\mathcal{E}_{\mathrm{L}}$	
		8-10	10-12	8-10	10-12	8-10	10-12
JOV	p1	99%	14%	88%	14%	70%	13%
	p2	34%	25%	31%	24%	28%	13%
	p3	54%	3%	48%	3%	50%	−4%
NPP	p1	27%	26%	28%	20%	28%	6%
	p2	50%	26%	41%	20%	38%	6%
	p3	37%	−10%	18%	−13%	16%	−12%
HRC	p1	23%	11%	–	–	–	–
	p2	33%	13%	–	–	–	–
	p3	43%	−6%	–	–	–	–
SS	p1	−11%	2%	–	–	–	–
	p2	−4%	8%	–	–	–	–
	p3	−13%	12%	–	–	–	–

Table 3.8: Percentage efficiency calculated from the global error in position, the relative error in Hamiltonian and the angular momentum, all for a local error tolerance 10^{-10}.

At tolerance 10^{-10}, 10-12-p1 and 10-12-p2 are more efficient than using tolerance 10^{-13} for the problem.

Figures 3.18 and 3.19 give the global error growth as a function of time using the 8-10 and 10-12 pairs respectively, when applied to Saturnian Satellites. Among the 8-10 pairs, only 8-10-p1 is efficient up to 17% at small tolerance as can be seen in Figure 3.18. No pair was found to be more efficient than 8-10-edp at the bigger tolerance of 10^{-10}. Referring to Figure 3.19 , all new pairs behave more efficiently than 10-12-dep at small tolerance. The pair 10-12-p3 being the most efficient yielding an efficiency of 24%, while 10-12-p1 and 10-12-p2 give 21% and 15% more efficiency than 10-12-dep. For tolerance 10^{-10}, these pairs could not increase in efficiency as the general trend was seen in the Jovian and Nine Planets Problems. (see Table 3.8.)

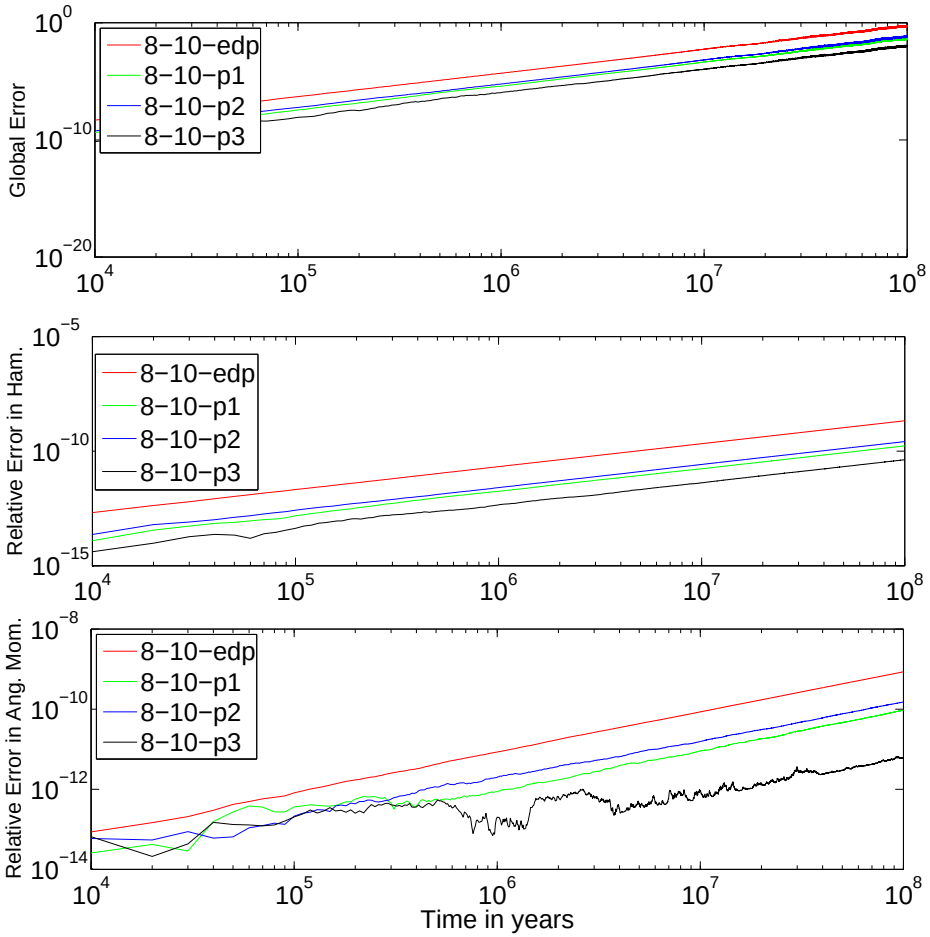

Figure 3.11: The error growth for the 8-10 pairs applied to the Jovian Problem over hundred million years for a local error tolerance 10^{-13}. (Top) Global error in position. (Middle) Relative error in Hamiltonian. (Bottom) Relative error in angular momentum.

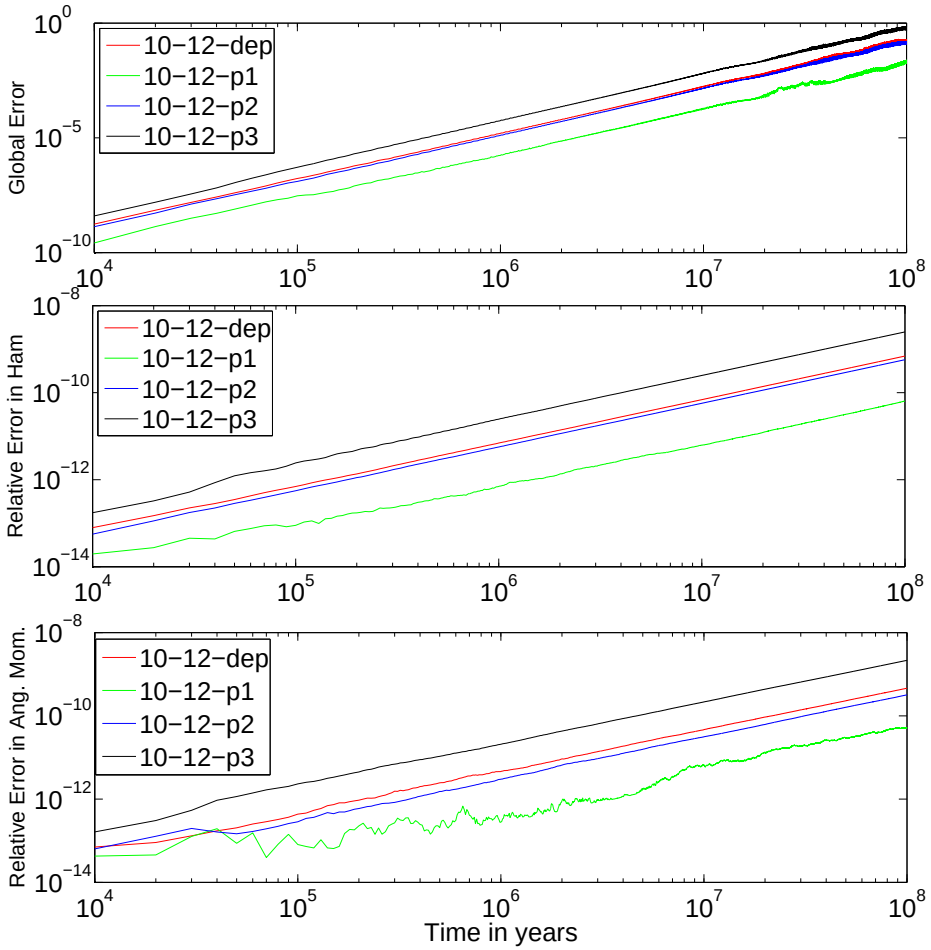

Figure 3.12: The error growth for 10-12 pairs applied to the Jovian Problem over hundred million years for a local error tolerance 10^{-13}. (Top) Global error in position. (Middle) Relative error in Hamiltonian. (Bottom) Relative error in angular momentum.

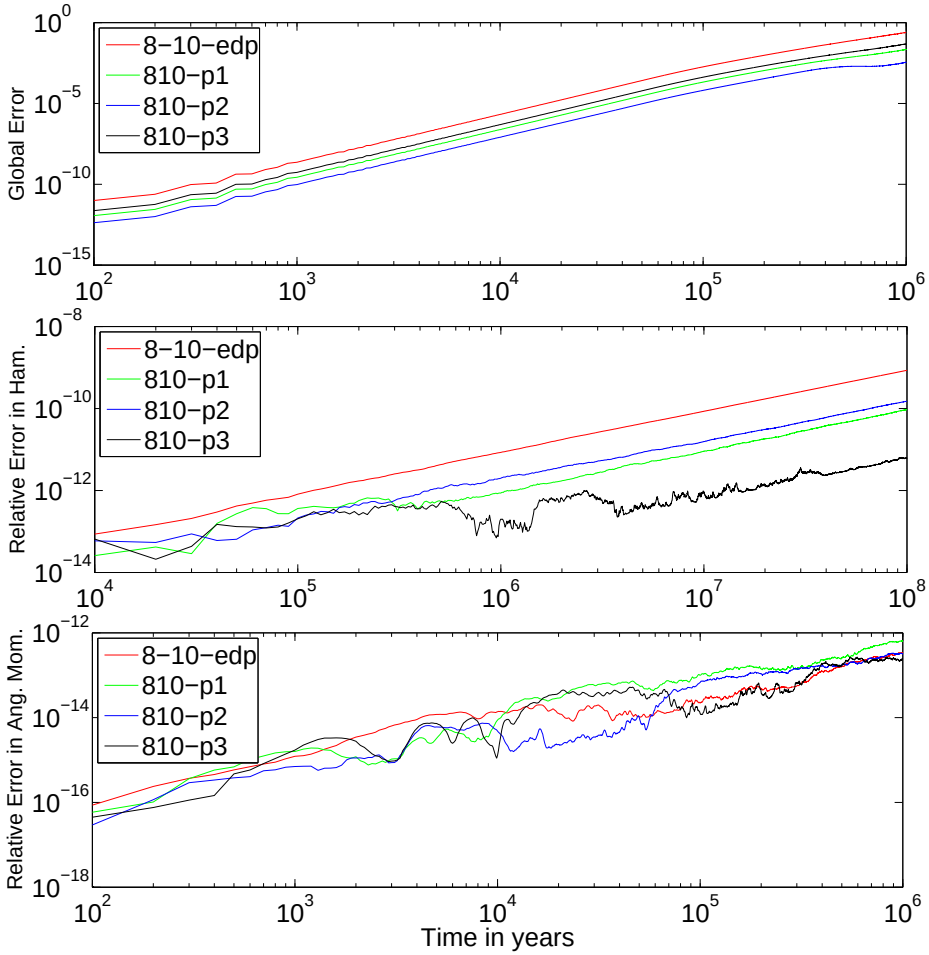

Figure 3.13: The error growth for 8-10 pairs applied to the Nine Planets Problem over one million years for a local error tolerance 10^{-13}. (Top) Global error in position. (Middle) Relative error in Hamiltonian. (Bottom) Relative error in angular momentum.

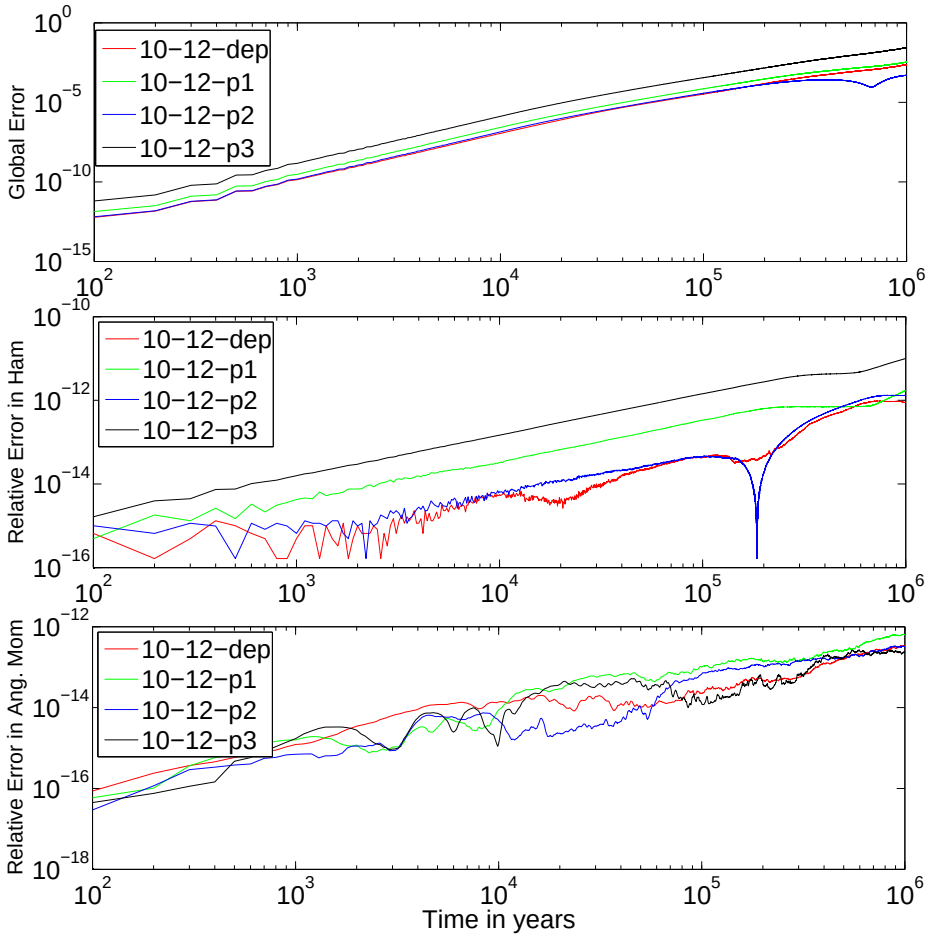

Figure 3.14: The error growth for 10-12 pairs applied to the Nine Planets Problem over one million years for a local error tolerance 10^{-13}. (Top) Global error in position. (Middle) Relative error in Hamiltonian. (Bottom) Relative error in angular momentum.

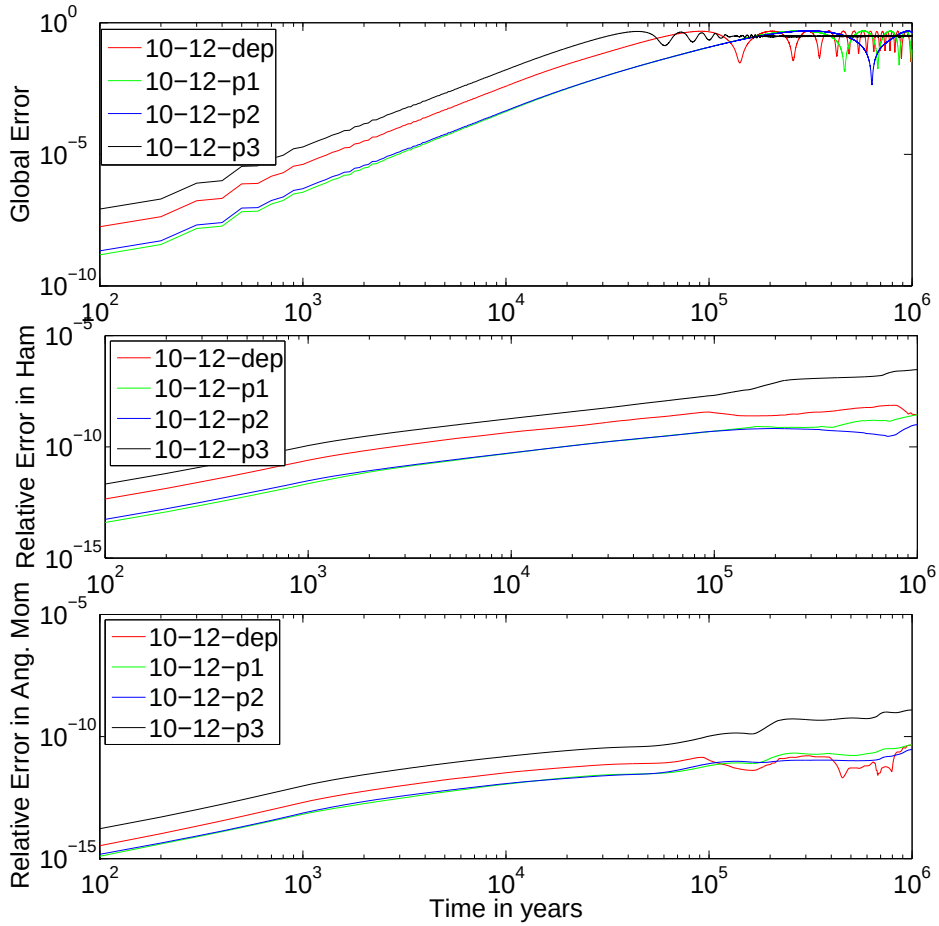

Figure 3.15: The error growth for 10-12 pairs applied to the Nine Planets Problem over one million years for a local error tolerance 10^{-10}. (Top) Global error in position. (Middle) Relative error in Hamiltonian. (Bottom) Relative error in angular momentum.

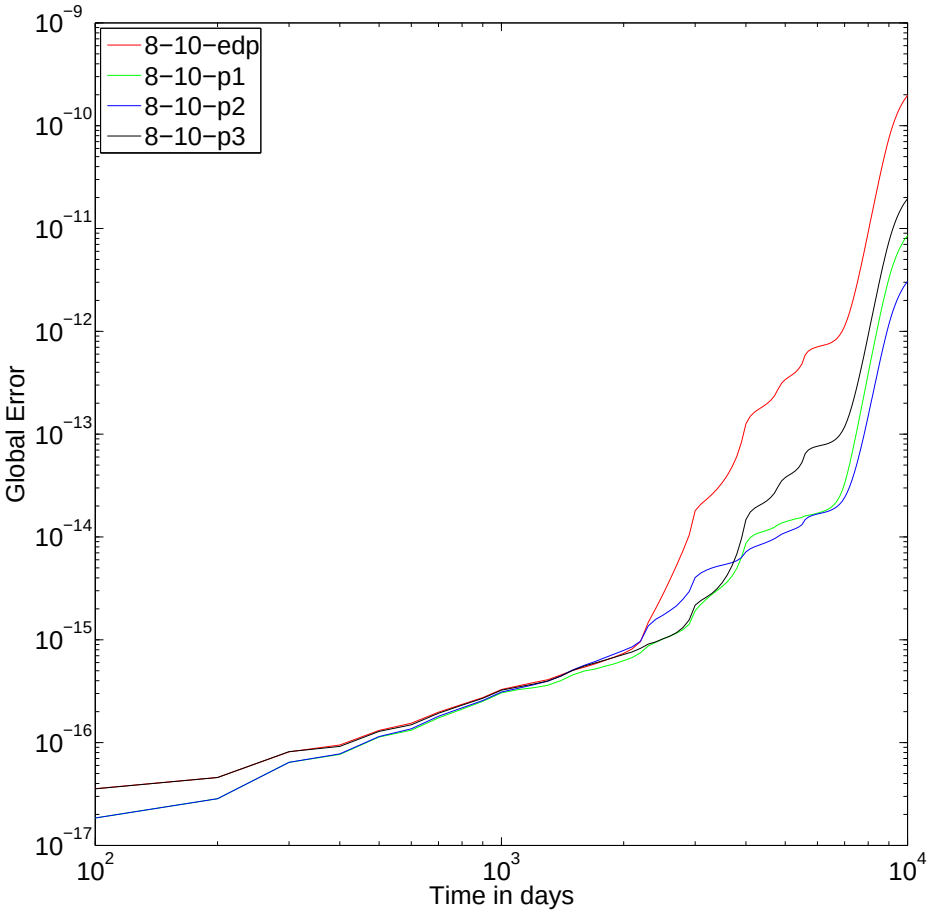

Figure 3.16: The error growth in position for 8-10 pairs applied to the HRC Problem over ten thousand days for a local error tolerance 10^{-13}.

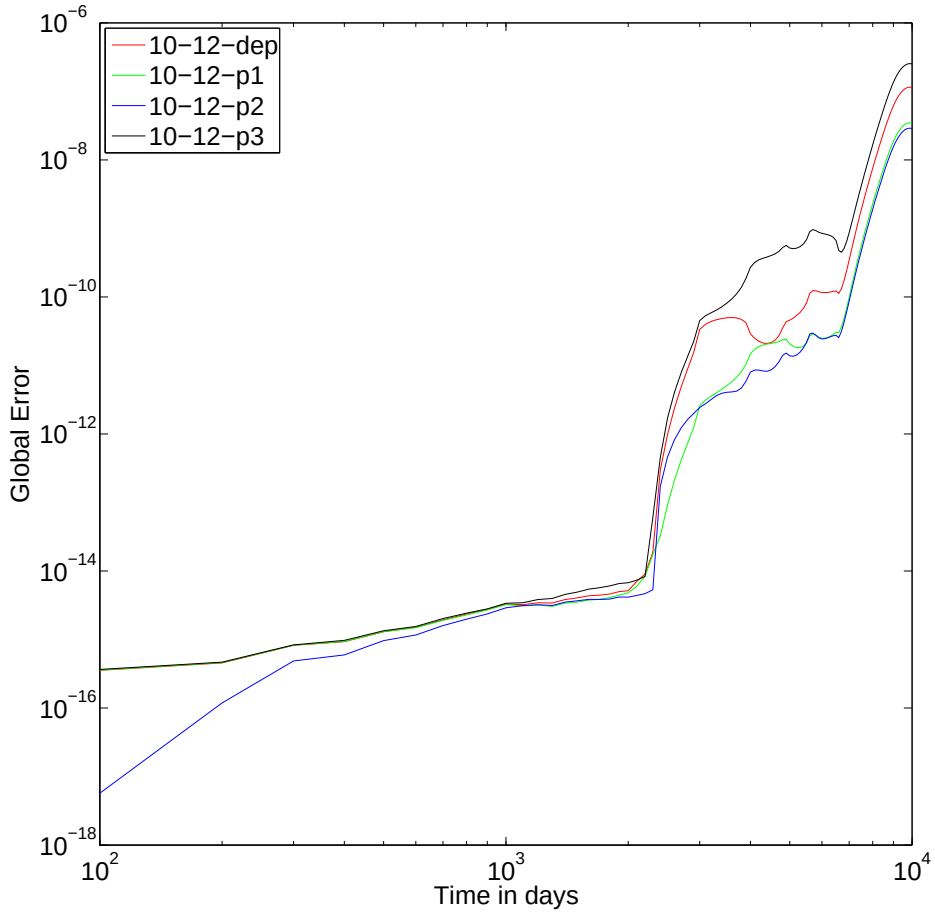

Figure 3.17: The error growth in position for 10-12 pairs applied to the HRC Problem over an interval of ten thousand days for a local error tolerance 10^{-10}.

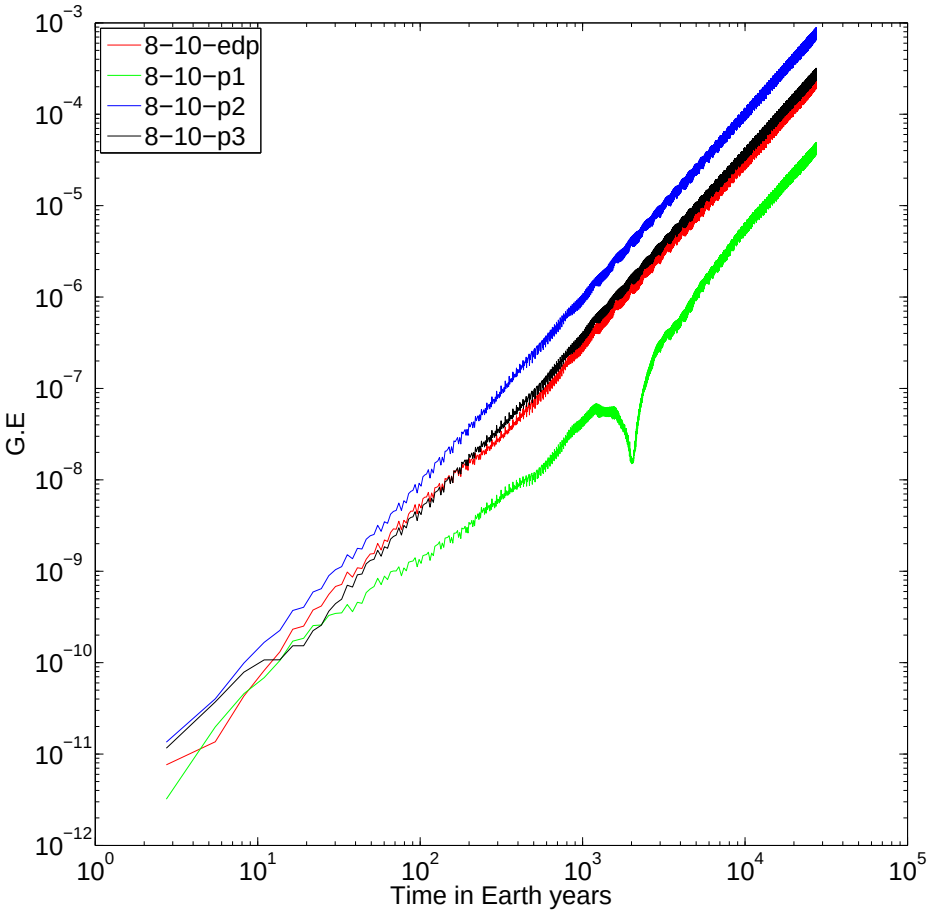

Figure 3.18: The growth of global error in position for 8-10 pairs applied to the Saturnian Satellites over an interval of 27 thousand years for local error tolerance 10^{-13}.

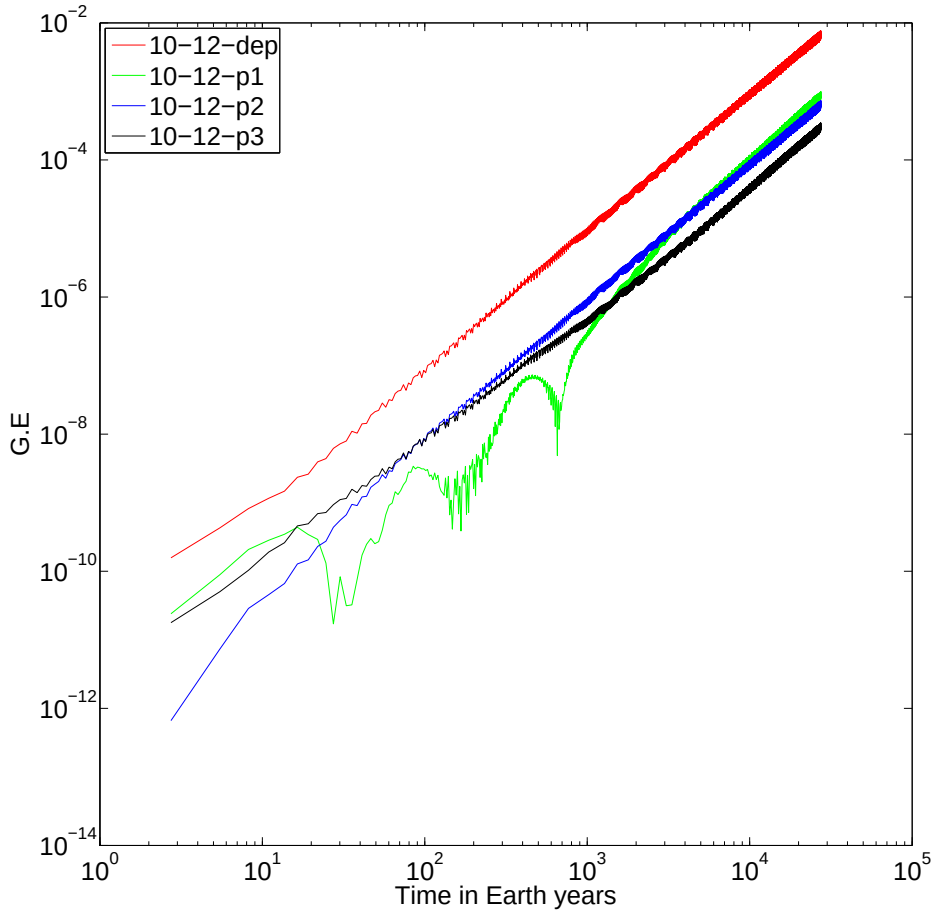

Figure 3.19: The growth of global error in position for 10-12 pairs applied to the Saturnian Satellites over an interval of 27 thousand years for a local error tolerance 10^{-13}.

3.9 Summary

The goal of this chapter was to find new high-order explicit Runge–Kutta Nyström pairs that were more efficient than existing pairs. We first presented a summary of how the order conditions for 8-10 and 10-12 pairs can be solved. We then found optimal 8-10 and 10-12 pairs by solving a constrained optimisation problem. The objective function was the norm of the leading error coefficients and the constraints were an upper bound on the magnitude of the coefficients, and lower bounds on the size of the stability regions. We found optimal pairs for different upper and lower bounds and tested these pairs on four long simulations in double precision for severe local error tolerances. Our testing showed that the best of the new 8-10 pairs was on average ten percent more efficient than the 8-10 pair of El-Mikkawy [26], and that the best of the 10-12 pairs were on average six percent more efficient than the 10-12 pair of Dormand *et al.* [21].

4

Achieving Brouwer's Law

4.1 Introduction

The numerical error in an approximate solution consists of truncation error and round-off error. For non-chaotic systems, Brouwer [6] showed for fixed step-size schemes that if the step-size was smaller than a prescribed value, random round-off error grows as the power law $x^{1/2}$ for conserved quantities such as total energy and angular momentum, and as $x^{3/2}$ for other dynamical variables such as the coordinates of particles. The growth is often called Brouwer's Law in literature [37, 45]. This growth contrasts with that when the round-off error is systematic. The power laws are then x and x^2 respectively.

Table 4.1 illustrates the growth of the round-off error for the power laws (ax^b) described above. The entries in the table are maximum of error for different values of exponent b in power law. We have assumed that the constant of proportionality in the

Time (years)	$x^{1/2}$	x	$x^{3/2}$	x^2
10^3	2×10^{-15}	2×10^{-14}	7×10^{-10}	2×10^{-9}
10^6	2×10^{-13}	3×10^{-12}	2×10^{-7}	3×10^{-5}
10^9	2×10^{-10}	7×10^{-9}	2×10^{-3}	—

Table 4.1: Illustrative round-off error for a one billion year integration of the Jovian Problem with two different error growth rates.

power laws is one.[1] We observe from the table that after one billion years of systematic round-off error there are no significant digits left in the position, and that when the round-off error is random there is approximately two significant digits. That is why a lot of interest has been shown in developing methods that have an error growth as $x^{3/2}$.

At least three integration schemes which achieve Brouwer's Law have been developed. The first scheme was due to Grazier *et al.* [37]. They implemented an order-13 Störmer method with the step-size chosen so that the truncation error was below machine precision. This choice meant that the only contribution to the numerical error was round-off error. Grazier *et al.* showed for the Jovian Problem, see [37, 38], that the phase error and error in energy grew as approximately $x^{3/2}$ and $x^{1/2}$ respectively when using the step-size of 4.1 days. Laskar *et al.* [65] presented a symplectic method of order $O(h^8\epsilon) + O(h^4\epsilon^2)$, where ϵ is a planetary mass in solar masses. They performed a simulation of the Sun, the eight planets, Pluto and the Moon using a step-size of 1.83 days and found the error in the energy satisfied Brouwer's Law. Hairer *et al.* [45] showed that when implicit Gauss Runge–Kutta (IRK) methods were implemented in the standard way Brouwer's Law was not achieved and that it was possible to modify the implementation so that Brouwer's Law could be achieved.

In this chapter we present comparisons between the IRK methods of Hairer *et al.* [45] and the Störmer methods of Grazier *et al.* [37]. We include the explicit Runge–Kutta Nyström 10-12 pair of Dormand *et al.* [21] to permit a comparison with a method that does not achieve Brouwer's Law. In addition, we investigate if the IRK methods can be made more efficient by using a higher degree polynomial for the initial estimate of the stage values (Hairer *et al.* [45] used a linear polynomial).

[1] Our numerical experiments on the Jovian Problem showed for one-step methods that the constant of proportionality was approximating one.

4.2 Implicit Runge–Kutta methods

IRK methods have the general form

$$y_{n+1} = y_n + h\sum_{i=1}^{s} b_i K_i, \tag{4.2.1}$$

where

$$K_i = f(x_n + c_i h,\ y_n + h\sum_{j=1}^{s} a_{ij} K_j), \quad i = 1, 2, \ldots, s.$$

The K_i are defined implicitly in terms of one another and must be found using an iterative scheme. Fixed point iteration and Newton's method are the most commonly-used schemes. Fixed point iteration is used for non-stiff problems and Newton's method for stiff problems.

Hairer *et al.* [45] used Gauss Runge-Kutta methods of four and six stages in their work. These methods have order eight and twelve, and we denote them by IRK8 and IRK12 respectively.

4.2.1 Achieving Brouwer's Law with IRK methods

The models used by Hairer *et al.* [45] for their simulations of the Solar System were non-stiff and hence Hairer *et al.* [45] used fixed point iteration to solve for the K_i. On each step, the system of non-linear equations $K - g(K) = 0$, where $K = [K_1^T, \ldots, K_s^T]^T$ and

$$g = [K_1^T - f^T(x_n + c_1 h,\ y_n + h\sum_{j=1}^{s} a_{1j} K_j), \cdots, K_s^T - f^T(x_n + c_s h,\ y_n + h\sum_{j=1}^{s} a_{sj} K_j)]^T, \tag{4.2.2}$$

is solved using the equation

$$K^{[m+1]} = g(K^{[m+1]}), \quad m = 0, 1, 2, \ldots, \tag{4.2.3}$$

where m is the number of iterations. This process is continued until the desired convergence is secured. The convergence of the above process depends on the function g and starting point K_0. Hairer *et al.* [45] proposed that, instead of using the usual fixed-point

convergence criteria, the iterations should be continued until the difference between the two consecutive iterations was below machine precision ϵ, ($\epsilon \approx 2.2 \times 10^{-16}$)

$$\max_{i=1,\ldots,s} \|K_i^{[m+1]} - K_i^{[m]}\| \leq \epsilon, . \tag{4.2.4}$$

They also found some systematic error contributing to the Hamiltonian due to inexact (rounded) coefficients of Gauss IRK. They avoided this contribution by representing the coefficients in higher precision.

In our first set of tests with the Gauss IRK methods, we used Hairer's code [43] to integrate the Jovian Problem for one million years and Kepler's two-body problem for 1000, 10,000 and 100,000 periods with different eccentricities up to 0.5. The reference solution for the Jovian Problem was found using a very accurate integration in quadruple precision, and the reference solution for Kepler's problem was taken from the exact solution as discussed in Chapter 1.

We solved the Jovian Problem for a large number of step-size ranging from 25 to 350 days. Figure 4.1 depicts the graph of the maximum relative error in the Hamiltonian across the interval of one million years as a function of the step-size. The solid blue line is for IRK8 and the solid red line for IRK12. We observe from the figure that both graphs have a tick (✓) shape. For larger step-size the truncation error dominates the round-off error and the relative error in the Hamiltonian behaves as the power law h^q where q is approximately the order of the integration method (as seen from Figure 4.1). For smaller step-size, the round-off error dominates the truncation error and the numerical error increases slowly as the step-size decreases. We refer to the step-size at which the numerical error is minimised as the optimal step-size. We observe from Figure 4.1 that the optimal step-size for IRK8 is approximately 80 days and that for IRK12 is approximately 185 days.

Another way of estimating the optimal step-sizes is to fit the least square lines to the increasing and decreasing parts of each graph and find the intersection of the lines. These lines are represented by the grey dashed lines in Figure 4.1. We observe from the figure that the intersection is at a step-size of 78 days for IRK8 and 187 days for IRK12, in good agreement with the previous estimates. Hairer *et al.* [45] used a step-size of 165 days in their experiments with IRK12.

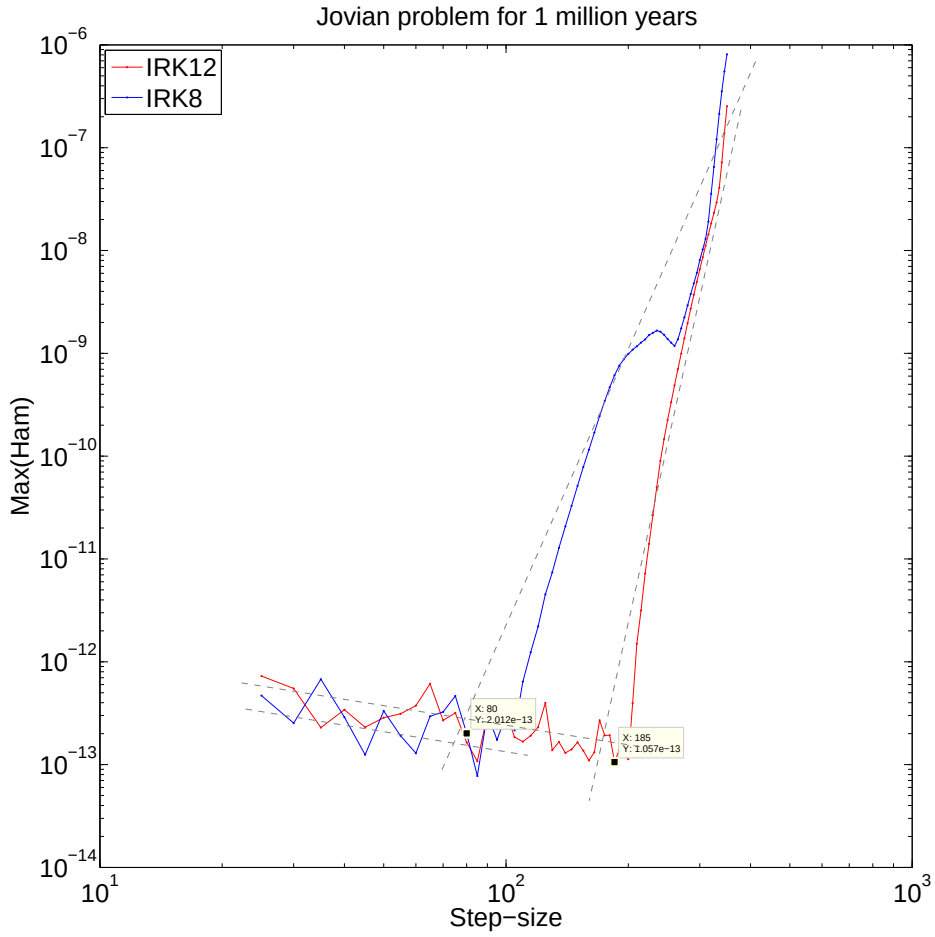

Figure 4.1: The maximum error in the Hamiltonian for step-sizes ranging from 350 days to 25 days for one million years of the Jovian Problem.

Figure 4.2 demonstrates the error growth rate in position using an optimal step-size for Jovian planets against the time, over an interval of one million years. The error was sampled every 10^4 years, hence the first point on the graph is at 10^4 years. The red and blue lines show the error for IRK12 and IRK8 respectively. The graphs have been smoothed in the same way as those in Section 3.8. We have added the grey dotted line to show the power law growth of $x^{1.5}$. We used linear least squares to fit the power law ax^b for global error and found that b was 1.44 and 1.32 for IRK12 and IRK8 respectively. These are in reasonable agreement with the theoretical value of 1.5 for stochastic error growth.

The exponents b of power law for global error and error in Hamiltonian are not 3/2 and 1/2. This suggests that the methods do not satisfy Brouwer's Law. However, the exponents of 3/2 and 1/2 are expected values and will only be achieved when an average over a suitable number of simulations with slightly different initial conditions is calculated.

Hairer *et al.* [45] demonstrated the random nature of the error growth for their model of the outer Solar System by performing the integration using 500 sets of perturbed initial conditions. We used the same procedure with a step-size of 185 days and integrated over an interval of 300,000 years (the size of the interval was a compromise between using as long an interval as possible and keeping the total CPU time requirement to an acceptable level).

The upper plot in Figure 4.3 illustrates the random walk of the error in the Hamiltonian for the perturbed initial conditions. In this figure, the relative error in the Hamiltonian as a function of time is shown for 100 perturbed initial values chosen randomly out of 500 perturbed initial conditions. The solid red lines show the average and standard deviation for the data. The average and standard deviation at the end of the integration were $\mu = 2.58 \times 10^{-15}$ and $\sigma = 7.52 \times 10^{-14}$ respectively. The average of the exponent for the power law growth of Hamiltonian was 0.52, in good agreement with Brouwer's Law.

The bottom plot of Figure 4.3 shows the histogram of the relative error in the Hamiltonian at $x = 300,000$ years. The solid blue line is the graph of the normal curve with the same mean and standard deviation as given above. We observe there is good agreement between the normal curve and the histogram.

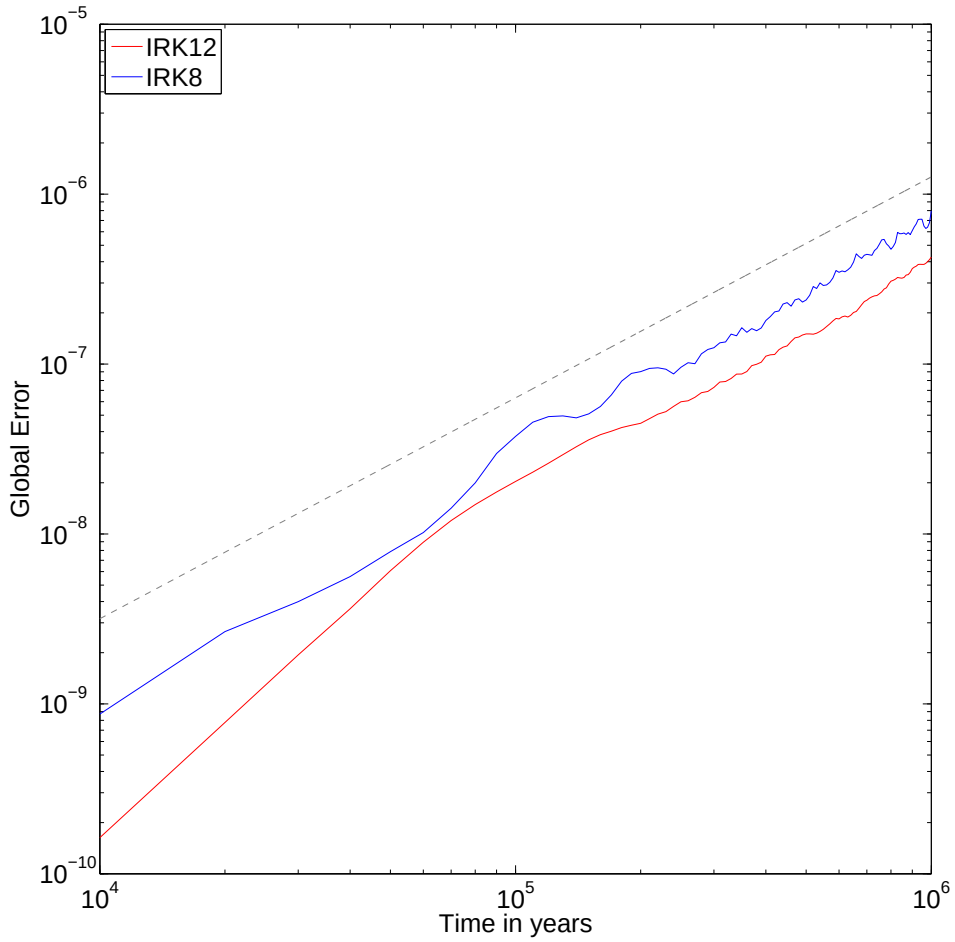

Figure 4.2: The error growth in the position of the planets for 1 million years of the Jovian Problem using IRK8 and IRK12.

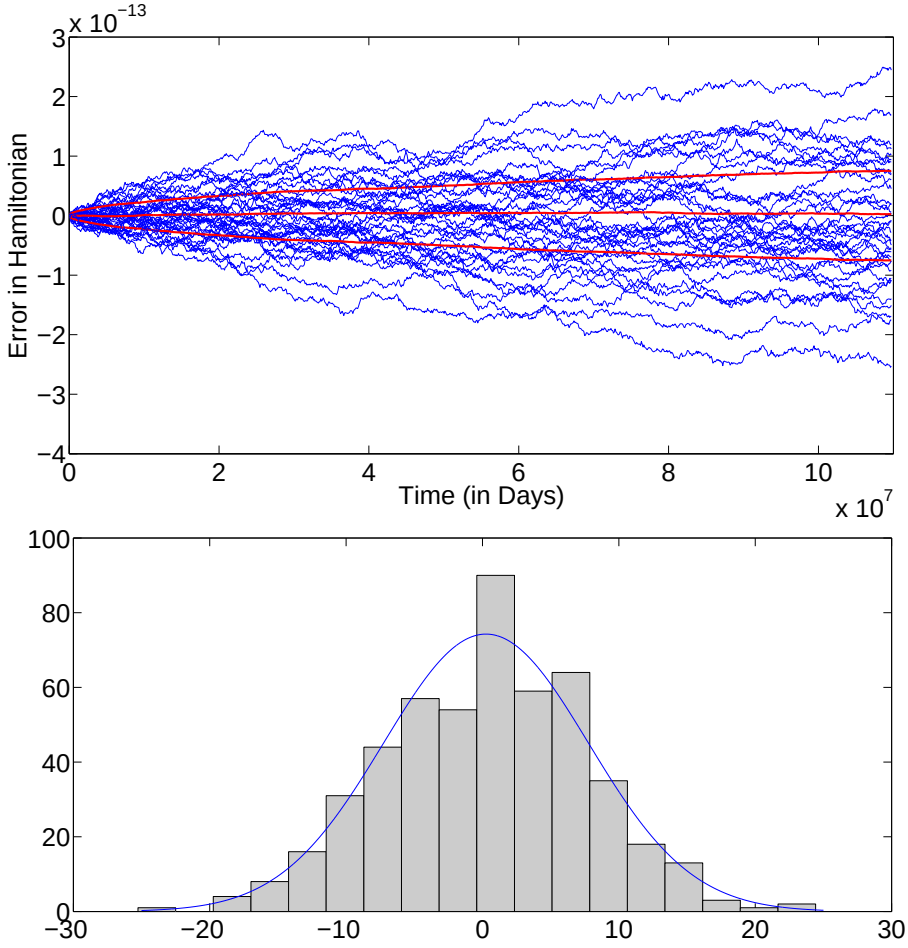

Figure 4.3: The error in Hamiltonian for 500 perturbed initial conditions for Jovian Problem. (Top) – Relative error in Hamiltonian for 100 randomly chosen perturbed initial values. (Bottom) – Histogram of Hamiltonian error at t=300,000 years against a normal distribution with the same mean and standard deviation. The horizontal axis is in units of 10^{-15}.

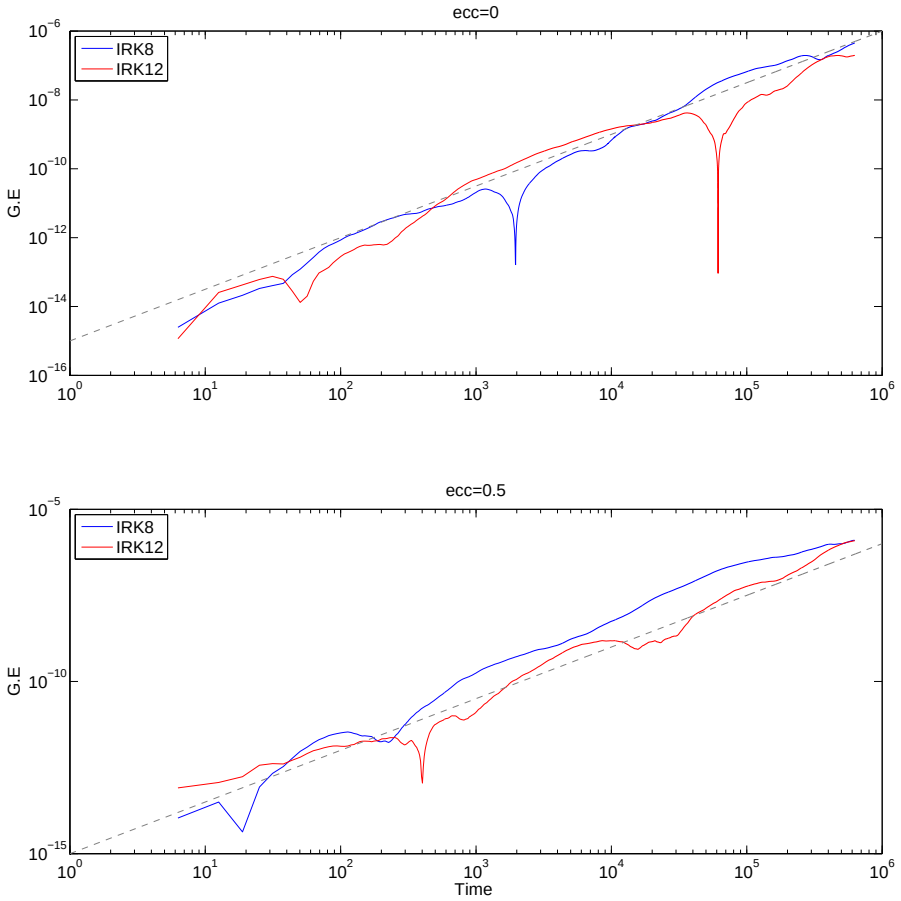

Figure 4.4: The error growth for Kepler's problem for 100,000 periods: (Top) – eccentricity is 0, (Bottom) – eccentricity is 0.5

.

We used the same way as for the Jovian Problem to find the optimal step-sizes for Kepler's problem. We found the optimal step-sizes of $\frac{2\pi}{44}$ and $\frac{2\pi}{105}$ for IRK8 with eccentricity of 0 and 0.5 respectively, and step-sizes $\frac{2\pi}{12}$ and $\frac{2\pi}{78}$ for IRK12 for the same eccentricities. Figure 4.4 gives the graphs of the global error as a function of x for Kepler's two-body problem with eccentricities of 0 and 0.5. The blue and red lines represent the global error growth for IRK8 and IRK12 respectively, the grey dotted line has slope 1.5 and is included for comparison purposes. Both methods have approximately the same global error at 100,000 periods for each eccentricity (see Figure 4.4).

4.2.2 Modification while implementing IRK

Hairer *et al.* [45] calculated the initial estimate for the argument of the stage value K_i by evaluating the linear polynomial

$$z_i = y_i + \alpha\ h\ f_i\ , \quad (y' = f), \tag{4.2.5}$$

at $\alpha = c_i$. We sought to reduce the number of function evaluations by using a higher degree polynomial based on past y values for the initial estimate.

A general form of n^{th} degree divided difference interpolation polynomial for given $(n+1)$ data points $(x_i, y_i),\ i = 0, 1, 2, \ldots, n$, can be expressed as

$$P_n(x) = y_1 + \sum_{i=1}^{n} d_{ii} \prod_{j=0}^{i-1} (x - x_j), \tag{4.2.6}$$

where $d_{ii},\ i = 1, \ldots, n$, are divided differences.

This polynomial can also be expressed in a simplified form when $x_i,\ i = 0, 1, \ldots, n$, are equally spaced. If we define $x_{i+1} - x_i = h$ and $x = x_0 + sh$, the above polynomial (4.2.6) takes the form

$$P_n(x) = y_1 + \sum_{i=1}^{n} d_{ii} s(s-1)\cdots(s-i+1)h^i.$$

This is called the Newton forward difference form. If the data points are recorded as $x_n, x_{n-1}, \ldots, x_0$ and are equally spaced with $x = x_n + sh$ and $x_i = x_n - (n-1)h$, the polynomial can be written in the Newton backward difference form

$$P_n(x) = y_n + \sum_{i=1}^{n} d_{ii} s(s+1)\cdots(s+i-1)h^i.$$

To make the method numerically stable and computationally efficient, Horner's algorithm, also called nested multiplication, is used to evaluate the above expression.

The polynomial is extrapolated at $x_n + c_i h\ , i = 1, \ldots, s$, and the values obtained

are used as starting points for the scheme. The extrapolated value at $x = x_n + c_i h$ is

$$P_n(x_n + c_i h) \;=\; y_n + c_i h(d_{n1} + (1 + c_i)h(d_{n2} + (2 + c_i)h(d_{n3} + \cdots + (n - 1 + c_i)h(d_{nn}))\cdots).$$

We found that changing the initial estimate from the linear polynomial (4.2.5) to a higher degree polynomial usually changed the optimal step-size. Hence, for each degree of the polynomial we had to re-estimate the optimal step-size. We did this in the same way as described in the previous subsection (4.2.1).

Tables 4.2 and 4.3 give the cost of integration for one million years of the Jovian Problem for polynomials up to degree 10 using IRK8 and IRK12. The acronym LP corresponds to linear polynomial that was used in [45], while $nDD,\ n = 2, \ldots, 10$, denotes the n^{th} degree polynomial used to get the initial values for the fixed point iteration. A positive percentage means that the new method was more efficient than LP while negative percentage means that the method was more expensive than LP.

The first row in Table 4.2 gives information about the integration using the linear polynomial of Hairer *et al.* [45]. The integration required approximately 49 million iterations and 200 million function evaluations, and took 249 seconds of CPU time (T_{cpu}). The L_2 norm of the maximum global error ($\mathcal{E}_{\text{ge}}$) was 1.21×10^{-6}.

The second row in the table gives information about the integration when the second degree polynomial is used for the initial estimate. The integration required 4.9% fewer iterations and 7.1% fewer function evaluations than for the linear polynomial. This resulted in a 9.4% reduction in the CPU time (the CPU time are accurate to one or two percent). On the downside, the norm of the maximum global error increased a small amount.

The third degree polynomial in the third row gives 7.4% and 9.5% fewer iterations and function evaluations respectively yielding 8% less CPU time required for the integration having similar global error.

When we used the fourth degree polynomial and a step-size of 80 days, we found that the global error was a lot larger than we expected. We experimented with other

Method	h	N_{it}	N_{fe}	T_{cpu}	Max($\mathcal{E}_{\text{ge}}$)	b
LP	80	48,814,577	199,808,308	249	1.21×10^{-6}	1.22
$2DD$	80	4.9%	7.1%	9.4%	1.40×10^{-6}	1.32
$3DD$	80	7.4%	9.5%	8.0%	1.22×10^{-6}	1.32
$4DD$	25	−89.3%	−84.9%	−84.4%	1.76×10^{-6}	1.54
$5DD$	14	−152.9%	−147.2%	−161.4%	2.02×10^{-6}	1.68
$6DD$	80	20.9%	22.5%	21.6%	7.57×10^{-7}	1.25
$7DD$	80	−12.1%	−9.3%	−4.7%	1.05×10^{-6}	1.20
$8DD$	70	22.5%	24.4%	20.6%	1.82×10^{-6}	1.64
$9DD$	80	−12.3%	−9.9%	−3.2%	9.65×10^{-7}	1.24
$10DD$	50	16.9%	15%	5.4%	7.05×10^{-7}	1.47

Table 4.2: A comparison of the polynomials for 1 million years of the Jovian Problem employing IRK8. The comparison is made using the optimal step-size h, number of iterations N_{it}, number of function evaluations N_{fe} and CPU time T_{cpu} .

step-sizes and found that a step-size of 25 days was optimal. We observe from the fourth row of the table that with this step-size, the integration requires about 84% more CPU time than for LP and the resulting error is larger, although the increase in the error is barely significant.

We observe from the rest of the table that an even degree polynomial led to less CPU time than LP and an odd degree polynomial to more CPU time. In all cases the global error was similar to that for LP.

Table 4.3 gives the results for IRK12. We observe that the optimal step-size varies considerably with the degree of the polynomial, and that the polynomials of degree four, five, seven, eight, nine and ten led to increased CPU time. The best degree was two and the reduction in CPU time was similar to that for the sixth degree polynomial used with IRK8.

We also performed simulations of the Nine Planets over 100,000 years. These simulations are done using optimal step-sizes, estimated as discussed earlier. They are 30 to 40 times smaller than those used for the Jovian Problem, in reasonable agreement with what we expect from the ratio of the smallest orbital period for each of the two problems. Table 4.4 gives the information on the cost of the integrations for IRK8. We observe that all degrees except four led to a reduction in the CPU time, and that the degree six polynomial is the most efficient among the polynomials that produced a global error similar

Method	h	N_{it}	N_{fe}	T_{cpu}	Max($\mathcal{E}_{\text{ge}}$)	b
LP	185	26,059,951	158,359,706	210	4.66×10^{-7}	1.38
$2DD$	250	19.2%	18.1%	19.1%	5.20×10^{-7}	1.64
$3DD$	200	8.9%	7.7%	7.6%	8.13×10^{-7}	1.42
$4DD$	25	−255%	−260%	−261%	8.26×10^{-7}	1.68
$5DD$	14	−376%	−382%	−418%	8.76×10^{-7}	1.63
$6DD$	185	10%	8.8%	9.5%	6.82×10^{-7}	1.58
$7DD$	185	−10.5%	−12%	−13.3%	2.59×10^{-7}	1.44
$8DD$	65	−49.8%	−51.7%	−53.6%	7.85×10^{-7}	1.56
$9DD$	100	−73.5%	−75.7%	−76.9%	3.22×10^{-7}	1.29
$10DD$	50	−52.6%	−54.6%	−68%	7.78×10^{-7}	1.57

Table 4.3: A comparison of the polynomials for 1 million years of the Jovian Problem employing IRK12. The comparison is made using the optimal step-size h, number of iterations N_{it}, number of function evaluations N_{fe} and CPU time T_{cpu} .

to that for LP.

Table 4.5 shows the integration cost for these polynomials using IRK12. We observe that using a polynomial other than LP was of no significant benefit neither in error reduction nor in CPU time.

4.3 Störmer methods

Störmer in 1907 [86] developed an accurate and simple method for solving (1.1.2) by adding the Taylor series for $y(x_n + h)$ and $y(x_n - h)$ and ignoring the higher order terms, see Hairer *et al.* [47], p. 462 for example. A Störmer method of order p can be written as

$$\begin{aligned} y_{n+1} - 2y_n + y_{n-1} &= h^2 \sum_{i=0}^{p-1} \alpha_i f_{n-i}, \\ y'_{n+1} - \frac{1}{h}(y_n - y_{n-1}) &= h \sum_{i=0}^{p-1} \beta_i f_{n-i}, \end{aligned} \tag{4.3.1}$$

where $f_{n-i} = y''_{n-i}$ and the coefficients α_i and β_i can be found using generating functions. The starting values $y_1, y_2, \ldots, y_n$, are usually computed using a one-step method. The method can also be written in terms of the backward difference interpolation polynomial

Method	h	N_{it}	N_{fe}	T_{cpu}	Max($\mathcal{E}_{\text{ge}}$)	b
LP	2	$206,209,305$	$843,100,020$	3407	8.6×10^{-6}	1.61
$2DD$	3	24.8%	24.6%	24.8%	2.2×10^{-5}	1.65
$3DD$	2	5.4%	5.6%	5.2%	5.7×10^{-6}	1.59
$4DD$	2	−5.2%	−5.3%	−4.2%	8.5×10^{-6}	1.60
$5DD$	2	10.8%	11.1%	10.5%	1.8×10^{-5}	1.72
$6DD$	2	21.4%	21.3%	21.1%	8.5×10^{-6}	1.61
$7DD$	2	15.5%	15.9%	15.2%	7.5×10^{-6}	1.60
$8DD$	2	17.1%	17.5%	15.9%	3.9×10^{-5}	1.75
$9DD$	2	18.2%	18.6%	17.3%	5.2×10^{-5}	1.76
$10DD$	2	18.8%	19.2%	17.4%	9.5×10^{-6}	1.63

Table 4.4: A comparison of the polynomials for 100,000 years of the Nine Planets Problem employing IRK8. The comparison is made using the optimal step-size h, number of iterations N_{it}, number of function evaluations N_{fe} and CPU time T_{cpu} .

Method	h	N_{it}	N_{fe}	T_{cpu}	Max($\mathcal{E}_{\text{ge}}$)	b
LP	6	$93,403,040$	$566,505,840$	2408	1.9×10^{-6}	1.55
$2DD$	6	−0.6%	−0.6%	−0.2%	7.8×10^{-6}	1.59
$3DD$	6	0.76%	0.77%	1.3%	4.5×10^{-6}	1.57
$4DD$	6	−4.9%	−4.9%	−4.5%	3.7×10^{-6}	1.56
$5DD$	6	1.5%	1.5%	1.7%	1.8×10^{-6}	1.54
$6DD$	6	1.3%	1.3%	1.4%	1.7×10^{-6}	1.54
$7DD$	3	−47.5%	−47.2%	−49.1%	8.1×10^{-6}	1.60
$8DD$	5	−10.1%	−10.5%	−35.6%	9.5×10^{-6}	1.63
$9DD$	3	−47.8%	−47.6%	−49.3%	7.9×10^{-5}	1.71
$10DD$	5	−11.8%	−11.2%	−37.4%	5.5×10^{-5}	1.74

Table 4.5: A comparison of the polynomials for 100,000 years of the Nine Planets Problem employing IRK12. The comparison is made using the optimal step-size h, number of iterations N_{it}, number of function evaluations N_{fe} and CPU time T_{cpu} .

passing through the points $(x_i, f_i),\ i = n-p+1, \ldots, n$. This gives

$$\begin{aligned} y_{n+1} - 2y_n + y_{n-1} &= h^2 \sum_{i=0}^{p-1} \gamma_i \bigtriangledown^i f_n, \\ y'_{n+1} - \frac{1}{h}(y_n - y_{n-1}) &= h \sum_{i=0}^{p-1} \sigma_i \bigtriangledown^i f_n, \end{aligned} \tag{4.3.2}$$

where $\bigtriangledown^0 f_n = f_n$ and $\bigtriangledown^{i+1} f_n = \bigtriangledown^i f_n - \bigtriangledown^i f_{n-1},\ i = 0, 1, \ldots,$
and

$$\gamma_i = (-1)^i \int_0^1 (1-s) \left(\binom{-s}{i} + \binom{s}{i} \right) ds,$$

$$\sigma_i = (-1)^i \left(\int_{-1}^0 (1+s) \binom{-s}{i} ds + \int_0^1 \binom{-s}{i} \right) ds.$$

The γ_i can be calculated from the recurrence

$$\gamma_i = 1 - \frac{2}{3} d_2 \gamma_{i-1} - \cdots - \frac{2}{i+2} d_{i+1} \gamma_0,$$

where $\gamma_0 = 1$ and $d_i = 1 + 1/2 + \cdots + 1/i$.

4.3.1 Störmer methods achieving Brouwer's Law

Grazier *et al.* [36] implemented Störmer methods in a way that achieved Brouwer's Law to within the uncertainty of the numerical experiments. Their implementation which they refer to as significance ordered computation has the following three important parts:

i. The coefficients in (4.3.1) become larger and are alternate in sign, as the order increases. This causes significant round-off error growth. Grazier *et al.* [36] implemented the Störmer methods in backward difference form as in (4.3.2). In the backward difference form (4.3.2), the coefficients γ_i and σ_i are all positive and monotonically decrease very slowly, thus reducing the round-off error.

ii. In addition Grazier *et al.* [36] avoided multiplying y_n by 2 in the first equation of (4.3.2) by using the technique named as "summed form". This practice is also formulated in [44] for the Störmer-Verlet scheme. Writing the first equation of (4.3.2) as $y'_{n+1/2} - y'_{n-1/2} = hf(y_n)$ and using $y'_{n+1/2} = (y_{n+1} - y_n)/h$ and $y'_{n-1/2} =$

$(y_n - y_{n-1})/h$, one gets

$$\begin{aligned} y'_{n+1/2} &= y'_{n-1/2} + hf(y_n), \\ y_{n+1} &= y_n + hy'_{n+1/2}. \end{aligned} \tag{4.3.3}$$

iii. Furthermore Grazier *et al.* [36] suggested using the insertion method when evaluating (4.3.2), see for example Higham [50]. In this method the values of a series are sorted in increasing magnitude and are also summed pairwise. So when calculating $\sum_{i=0}^{p-1} \gamma_i \bigtriangledown^i f_n$, the f_i are sorted such that $|f_{n-i}| < |f_{n-i+1}| < \cdots < |f_n|$ and also summed in the same manner but in the form of pairs as $(\cdots((f_{n-i} + f_{n-i+1}) + f_{n-i+2}) + \cdots + f_n)$. In practice, the backward differences decrease in magnitude with increasing i and it is sufficient to sum backwards over i.

Grazier *et al.* [36] then chose the step-size so that the truncation error was below machine precision.

4.4 Comparisons

We compared the IRK and Störmer methods described in the previous sections, and the ERKN 10-12 pair of Dormand *et al.* [21] on the Jovian Problem over an interval of 10^8 years and the Nine Planets Problem over 10^5 years. The emphasis in our comparison is on the accuracy of the solution including the phase information.

Rather than using the same variant of the above IRK methods for all problems, we used the most efficient variant for each problem. This choice meant that the IRK methods were shown in the best light. For the Störmer methods, we used order 12, 13 and 15 - these methods are denoted by the acronyms S12, S13 and S15 respectively. Order 13 was used because this order was recommended by Grazier [35]. Order 12 and 15 were used to illustrate the dependance on the order. The Störmer methods were implemented in Fortran in a way similar to that of N-body integrators NBI developed by Varadi [92]. All the comparisons are performed in double precision.

4.4.1 Jovian Problem

We used IRK8-6DD with a step-size of 80 days and IRK12-2DD with a step-size of 250 days. The step-size used for the Störmer method was 4 days, a value very similar to that used by Grazier *et al.* [36]. The tolerance for the ERKN pair was 10^{-14} which is the minimum usable tolerance in double precision arithmetic. This choice gave an average step-size of 200 days.

The global error in the position was estimated at N evenly spaced values on the interval of integration where N was either 1000 or 10,000. The reference solution for error estimation was calculated in quadruple precision using a tolerance of at least 10^4 times smaller than that used for double precision simulations. The quadruple precision simulations require approximately 100 times as much CPU time as the simulations in double precision.

Table 4.6 summarises the results of the integration for IRK8-6DD, IRK12-2DD, S13 and the ERKN 10-12 pair. The table lists the CPU time T_{cpu} , the L_2 norm of the maximum global error $\mathcal{E}_{\mathrm{ge}}$, and exponents b for the least squares fit of the power law ax^b to $\mathcal{E}_{\mathrm{ge}}$ and the relative error $\mathcal{E}_{\mathrm{H}}$ in the Hamiltonian for each method. We observe for the methods which satisfy Brouwer's Law that IRK12-2DD uses the least CPU time. We also observe that S13 requires four times as much T_{cpu} as IRK12-2DD and produces a solution for which $\mathcal{E}_{\mathrm{ge}}$ is over three times as large as that for IRK12-2DD.

If we now consider the performance of the ERKN 10-12 pair, the results clearly illustrate the trade-off between accuracy and CPU time. The 10-12 pair used only 20% of the CPU time of IRK12-2DD but $\mathcal{E}_{\mathrm{ge}}$ was 129 times as large. If our requirement when doing the simulation is to get the most accurate solution, the fact that the 10-12 pair needed far less CPU time would be irrelevant.

The exponents in the last two columns of the Table 4.6 for the IRK and Störmer methods are not 3/2 and 1/2. The reason for this is discussed in section 4.2.1. However, this reasoning does not apply to the ERKN pair because the round-off error is systematic and not random. Hence, we expect to get exponents close to the values of 2 and 1 respectively.

Method	h	T_{cpu}	Max($\mathcal{E}_{\rm ge}$)	$b($ $\mathcal{E}_{\rm ge}$ $)$	$b($ $\mathcal{E}_{\rm H}$ $)$
IRK8-6DD	80	22,704	3.56×10^{-3}	1.62	0.38
IRK12-2DD	250	18,695	1.17×10^{-3}	1.60	0.57
S13	4	74,739	3.76×10^{-3}	1.58	0.33
ERKN 10-12	200	3,721	1.51×10^{-1}	2.04	0.99

Table 4.6: A comparison of the methods for 100 million years of the Jovian Problem, at optimal step-size h, CPU time T_{cpu} in seconds, Maximum of global error in positions $\mathcal{E}_{\rm ge}$, exponent of power law for global error and relative error in Hamiltonian $\mathcal{E}_{\rm H}$.

The final observation we make from the table is that even when a method satisfies Brouwer's Law the accumulated error in a very long simulation can mean the position of the planets is poorly known. This is in agreement with the idealised results in Table 4.6.

4.4.2 Nine Planets Problem

The shortest orbital period in the Nine Planets Problem is 88 days (Mercury) as against the shortest orbital period of 4333 days (Jupiter) in the Jovian Problem. Hence we can expect the optimal step-sizes for the Nine Planets Problem to be approximately 50 times smaller than those for the Jovian Problem.

We found that IRK12-6DD was the most efficient among the variants of the IRK12 methods. The optimal step-sizes we used were 2, 6 and 0.08 days for IRK8-6DD, IRK12-6DD and S13 respectively. Table 4.7 summarises the results of the integrations. As with the Jovian Problem, IRK12-6DD required the least CPU time among the methods and had the smallest value of $\mathcal{E}_{\rm ge}$, although $\mathcal{E}_{\rm ge}$ for S13 was just 3% larger than that for IRK12-6DD. S13 required 3.6 times as much CPU time as IRK12-6DD. This is less than the factor of four for the Jovian Problem. One possible reason for this difference is that the cost per equation of evaluating the acceleration for the Nine Planets Problem is greater than that for the Jovian Problem.

The ERKN pair used only 30% of the CPU time of the IRK12-6DD but unlike in the Jovian Problem, the $\mathcal{E}_{\rm ge}$ is only 7 times as large. This may be because the Nine Planets Problem is integrated 10^3 times less than that done for the Jovian Problem. This reduction is not fully compensated by the smallest orbital period in the Nine Planets Problem being 50 times smaller than that for the Jovian Problem. In addition, we found

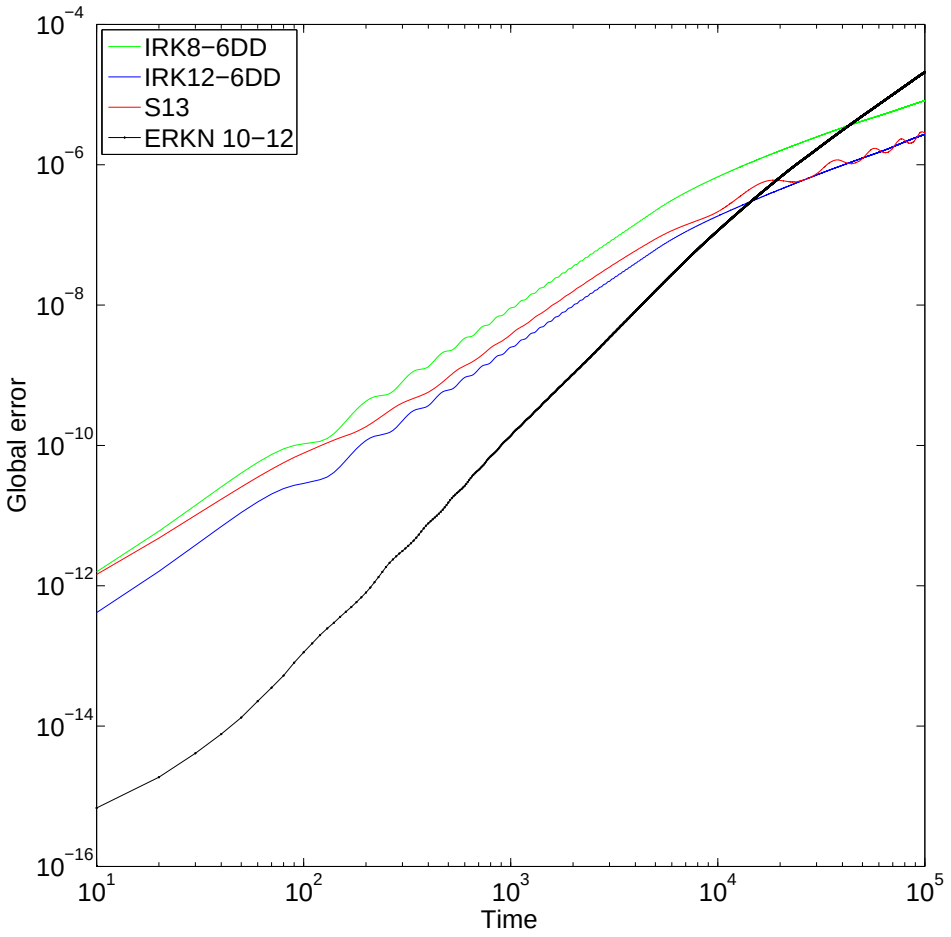

Figure 4.5: The error growth in the position for the Nine Planets Problem using the IRK, Stormer and ERKN methods over 100 thousand years.

that the global error for the 10-12 pair was smaller than that for the other methods up to approximately 2×10^4 years. For larger values of x, the quadratic growth of the global error for the the 10-12 pair means the error is larger than that for the other methods. This behaviour is clearly illustrated in Figure 4.5. The behaviour also occurred for the Jovian Problem, although the crossover point was at a different value of x.

4.5 Continuous extension

One-step methods are usually formulated to produce successive approximations y_n to $y(x_n)$ on the mesh points $x_0 < x_1 < x_2 < \cdots$. These mesh points are determined by a step-size selection strategy based on the methods. If the approximation to the solution is

Method	h	T_{cpu}	Max($\mathcal{E}_{\rm ge}$)	b($\mathcal{E}_{\rm ge}$)	b($\mathcal{E}_{\rm H}$)
IRK8-6DD	2	2880	8.12×10^{-6}	1.58	0.34
IRK12-6DD	6	2280	2.72×10^{-6}	1.35	0.56
S13	0.08	8160	2.80×10^{-6}	1.41	0.19
ERKN 10-12	4	660	2.11×10^{-5}	2.06	0.89

Table 4.7: A comparison of the methods for 100 thousand years of the Nine Planets Problem, at optimal step-size h, CPU time T_{cpu} in seconds, Maximum of global error in positions $\mathcal{E}_{\rm ge}$, exponent of power law for global error and relative error in Hamiltonian $\mathcal{E}_{\rm H}$.

required at $x = x^*,\ x^* \in [x_n, x_{n+1}]$, the approximate solution at the mesh points must be extended into a continuous approximation.

An important application of continuous approximations occurs when detecting if a small body hits a planet. It might be possible that a collision occurs within the step and not at the end of it. This collision would not be detected if we have the approximate solution at just the mesh points. One way to detect the collision is to use a far smaller step-size when the small body is near a planet than when the small body is far from the planet. This would be inefficient because all bodies including those that were not colliding with a planet would be integrated with the smaller step-size. This inefficiency can be avoided by extending the solution at the mesh points to a continuous approximation across the step and using root finding techniques to check for a collision. The continuous extensions we consider are often called interpolants. In this section, we investigate the use of interpolants for the IRK methods.

The very first work on interpolants for Runge-Kutta methods was done by Horn [52] and Shampine [79] for explicit methods. Enright [29], Dormand [23], Tsitouras [90] and Verner [94] extended this work. Less work has been done for IRK methods, see for example, the integrators RADAU5 and SDIRK4 [42, 46].

For an s-stage IRK method, the s intermediate values

$$Y_i = y_n + h \sum_{j=1}^{s} a_{ij} K_j, \qquad i = 1, \ldots, s,$$

are of order s. We can easily form an s^{th} degree polynomial $P_s(x)$ from the Y_i and the solution at the end of the step using a divided difference formulation. If we denote the divided difference coefficients of the polynomial by $d_{ii}, i = 0, \ldots, s$, the polynomial can

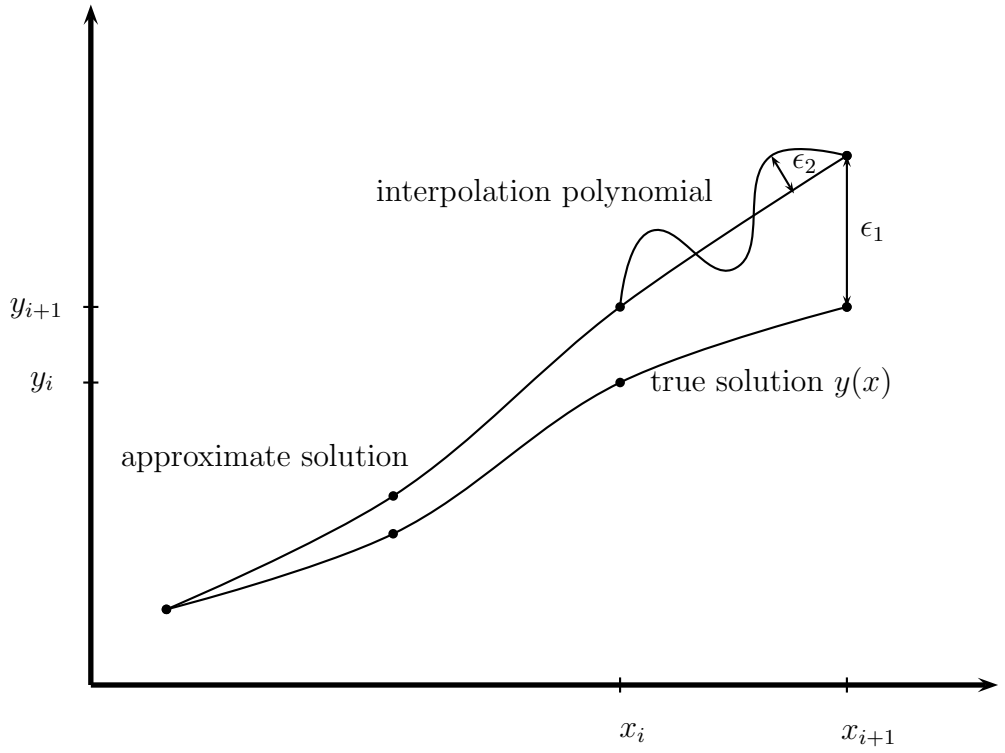

Figure 4.6: Classification of error using numerical method and interpolation polynomial.

be written as

$$P_s(x) = Y_1 + \sum_{i=1}^{s} d_{ii} \prod_{j=0}^{i-1} (x - x_j)h^i, \tag{4.5.1}$$

where

$$x_j = x_{n-1} + c_{j+1}h, \quad j = 0, 1, \ldots, s-1.$$

The order of the Gauss methods is $2s$ while the interpolation polynomial is of order s. So the error at the mesh points $x_0 < x_1 < x_2 < \ldots$ is much smaller than that of intermediate values. The total error in a value of y calculated using $P_s(x)$ is the sum of the error introduced by the IRK method and the polynomial, as illustrated in Figure 4.6. The error in the solution grows as $\epsilon_1 = Cx^{3/2}h^{2s}$ and the error introduced by the interpolation is $\epsilon_2 = Dh^{s+1}$. Hence the total error is $\epsilon_1 + \epsilon_2$. The ratio R_e of this error to the error at the mesh points is then

$$R_e = \frac{\epsilon_1 + \epsilon_2}{\epsilon_1} = 1 + \frac{\epsilon_2}{\epsilon_1} \tag{4.5.2}$$

This ratio tends to 1 as $x \to \infty$.

To test the above analysis we integrated the Jovian Problem over one million years using IRK8-6DD and IRK12-2DD with optimal step-sizes. On each step, we formed the interpolant and evaluated it at 10 evenly spaced values of x on the step. We then

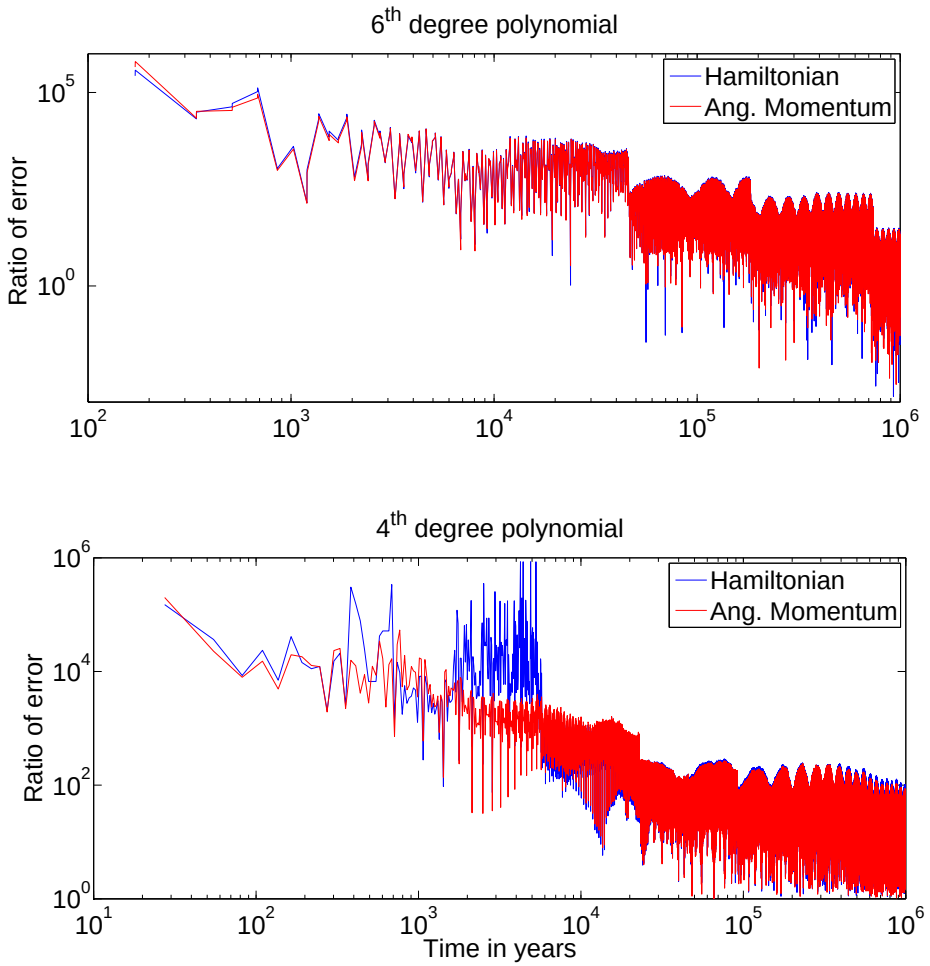

Figure 4.7: The ratio of the relative error in Hamiltonian and angular momentum for one million years of the Jovian Problem while implementing continuous extension for IRK methods. (Top) - sextic interpolant implemented on IRK12. (Bottom): quartic interpolant implemented on IRK8.

calculated the L_2 norm of the relative error in the Hamiltonian and angular momentum at each of these points and divided these errors by the corresponding error at the end of the step to give the ratio R_e defined above.

Figure 4.7 gives the graphs of R_e for IRK12-2DD and IRK8-6DD. The top half of the figure has the graph of the ratio for the error in Hamiltonian and angular momentum using IRK12-2DD with the 6^{th} degree interpolant. We observe that R_e for both the Hamiltonian and angular momentum is large for small x and decreases, with oscillations, as x increases, and approaches one, confirming our analysis above.

The bottom half of the figure depicts the ratio of the error in Hamiltonian and angular momentum for IRK8-6DD with the 4^{th} degree interpolant. We observe that the ratio for both the Hamiltinian H and angular momentum L decreases with time but have not approached one as closely as for IRK12. One possible reason the ratio is not one within the 10^6 years is that the order of polynomial is four rather than six.

The disadvantage of a large R_e for small x on the Jovian Problem does not occur with the implementation of Störmer methods given by Grazier *et al.* [36]. Graizer *et al.* [40] showed that when the step-size is chosen so that the methods satisfy Brouwer's Law, one-step quintic interpolation will be sufficiently accurate for all x.

To illustrate this difference between cubic and quintic, we added the cubic and quintic Hermite polynomials to our Störmer integrator and then solved the Jovian Problem in a similar manner for that of IRK8 and IRK12 using an optimal step-size of four days. For cubic, $y_n, y'_n, y_{n+1}, y'_{n+1}$ are needed to construct a polynomial while including y''_n and y''_{n+1}, quintic polynomial can be obtained. The polynomials as implemented in [40] are given by

$$P_3(x) = d_0 y_n + d_1 h y'_n + d_2 y_{n+1} + d_3 h y'_{n+1} \tag{4.5.3}$$

where $d_0 = (\tau - 1)^2(2\tau + 1)$, $d_1 = (\tau - 1)^2\tau$, $d_2 = (3 - 2\tau)\tau^2$, $d_3 = (\tau - 1)\tau^2$
and $\tau = (x - x_n)/h$.
The quintic interpolation polynomial is

$$P_5(x) = d_0 y_n + d_1 h y'_n + d_2 h y''_n + d_3 y_{n+1} + d_4 h y'_{n+1} + d_5 h y''_{n+1} \tag{4.5.4}$$

here $d_0 = (1 - \tau)^3(6\tau^2 + 3\tau + 1)$, $d_1 = (1 - \tau)^3\tau(3\tau + 1))$,
$d_2 = (1 - \tau)^3\tau^2/2$, $d_3 = \tau^3(6\tau^2 - 15\tau + 10)$, $d_4 = \tau^3(1 - \tau)(3\tau - 4)$, $d_5 = \tau^3(\tau - 1)^2/2$.

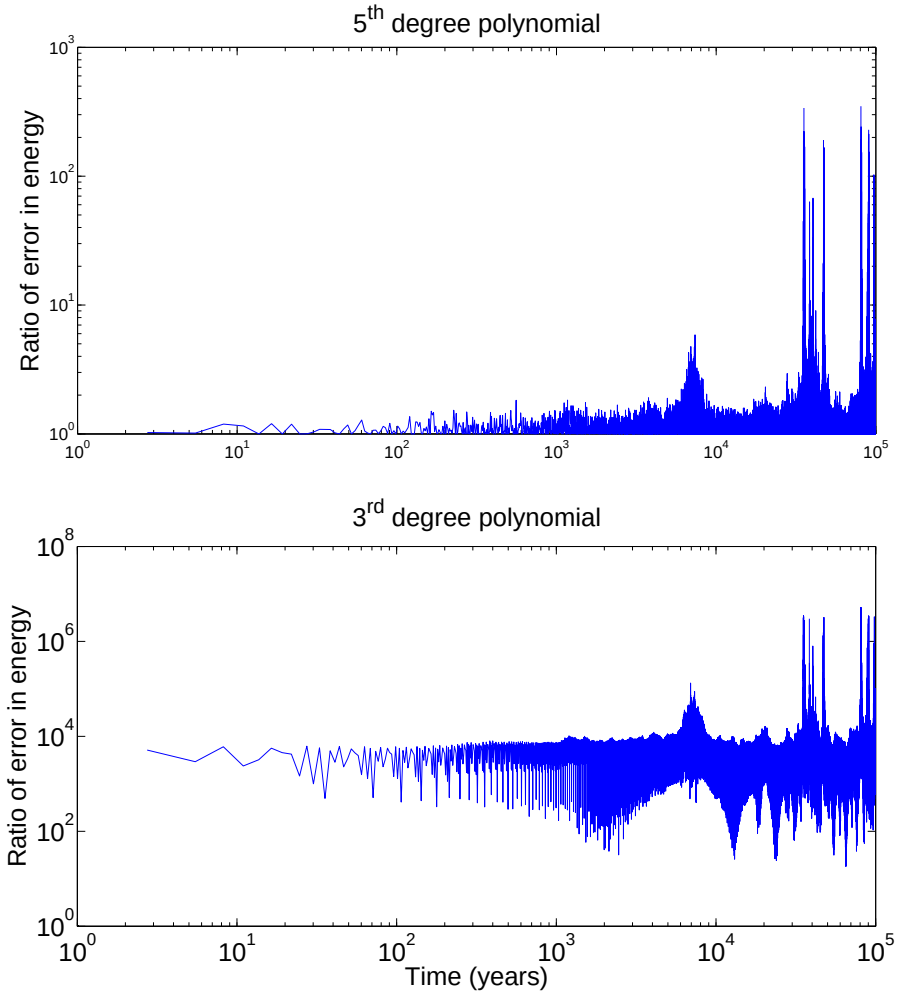

Figure 4.8: The ratio of the relative error in Hamiltonian for 100,000 years of the Jovian Problem while implementing continuous extension for Störmer method of order 13. (Top) - quintic interpolant implemented on Störmer method. (Bottom): cubic interpolant implemented on Störmer method.

We calculate the relative error in Hamiltonian for 100,000 years of the Jovian Problem. The error is calculated at each day using interpolation polynomial within the interval. Figure 4.8 contains the plots of the ratio R_e for the energy. The top plot is for quintic interpolation and the bottom is for cubic interpolation. We observe from the top plot that the ratio of error in Hamiltonian is approximately one up to 7×10^3 years and then oscillates at some points, as can be seen in Figure 4.8. The bottom plot in Figure 4.8 shows that using cubic polynomial the ratio of error in energy does not decrease as was expected. This verifies the results of Graizer *et al.* [40]. This may be because that the interpolation polynomial used for velocity coordinates is of order 2.

4.6 Summary

The chapter deals with efficient implementation of implicit Runge–Kutta methods, which achieve Brouwer's Law. We investigated that the IRK methods having order eight and twelve can be made more efficient by using a higher degree polynomial for the initial estimate of the stage values (linear polynomials are in use in the original method by Hairer *et al.* [45]). This gave us a significant decrease in the number of iterations and hence in the CPU time, which is worthwhile for long-term simulation of the Solar System. We also presented comparisons between the IRK methods of Hairer *et al.* [45] and the Störmer methods of Grazier *et al.* [37]. We included the explicit Runge–Kutta Nyström 10-12 pair of Dormand *et al.* [21] to permit a comparison with a method that does not achieve Brouwer's Law. The provision of continuous extension has been considered using interpolation polynomials for IRK methods. This is done by fitting a polynomial interpolant to the discrete IRK function evaluations. We used the 4^{th} and 6^{th} degree polynomials for eighth and twelve order methods respectively. These polynomials were implemented on Jovian problem and ratio of error at intermediate and end points of the interval was calculated. It was shown that the ratio becomes one as the time of integration proceeds.

5

Conclusions

The goal of this thesis was to obtain more efficient methods for performing accurate N-body simulations of the Solar System. We investigated two types of methods: higher order explicit Runge–Kutta Nyström (ERKN) pairs and implicit Runge–Kutta (IRK) methods used with a fixed step-size and fixed point iteration. Throughout the thesis, we used several realistic test problems. These consisted of the Jovian, Nine Planets, Helin-Roman-Crockett (HRC) and Saturnian satellites Problems. We also included Kepler's two-body problem in our test problems as it has an analytical solution.

We investigated the ERKN pairs of two classes having orders 8-10 and 10-12 by searching efficient pairs and then comparing them on our test problems. The objective function, while searching the coefficients for 8-10 and 10-12 pairs, is the leading error coefficient $\tau^{(11)}$ and $\tau^{(13)}$ respectively. We also used $\tau^{(13)}$ and $\tau^{(14)}$ as an objective function for the 8-10 and 10-12 pairs. We used Horn's and Matrix stability criteria as part of our constraints. These constraints are incorporated using penalty functions. We make use of the algorithm known as simulated annealing (SA) for our minimisation problem.

Simulated annealing is a direct search method based on optimisation. The minimisation of the objective function using SA leads to a large number of near optimal pairs. The pairs obtained by using Matrix stability criteria as constraint are not efficient than the 8-10 pair of El-Mikkawy [26] and 10-12 pair of Dormand *et al.* [21]. On the other hand, pairs obtained by keeping Horn's stability interval as one of the constraints are more efficient. Numerical experiments show that the extended stability intervals do not have a significant effect on the efficiency of the pairs. We also observe that pairs obtained by keeping $\tau^{(11)}$ for 8-10 pairs and $\tau^{(13)}$ for 10-12 pairs as objective function are more efficient than keeping $\tau^{(12)}$ and $\tau^{(14)}$ as objective functions for 8-10 and 10-12 pairs respectively.

We implemented the new8-10 and 10-12 pairs on test problems for short and long intervals of integration. Our numerical testing consists of two parts. In a preliminary testing, efficiency graphs are plotted at a range of tolerances for a short interval of time (to avoid an access amount of CPU time) using new pairs. Some of the new8-10 pairs become more efficient on lax tolerances and some on severe tolerances for 8-10 pairs, except for Saturnian satellite. whereas all the 10-12 pairs are giving better efficiency on severe tolerance. In the second part, we tested the more efficient 8-10 and 10-12 pairs from the first part on longer intervals of integration. The efficiency is measured at lax and severe tolerances keeping the same number of function evaluations. We find that the new 8-10 pairs are up to 56%, 62%, 57% and 17% more efficient for the Jovian, Nine Planets, HRC and Saturnian satellites problems respectively, when compared with the 8-10 pair of El-Mikkawy [26]. The new 10-12 pairs did not prove to be much more efficient as 8-10 pairs. We obtain that for Jovian, Nine Planets, HRC and Saturnian satellites, new pairs are 18%, 26%, 13% and 24% much more efficient than the 10-12 pair of Dormand *et al.* [21] respectively.

We also implemented the IRK methods achieving the optimal error growth. This optimal error growth is $x^{1.5}$ and $x^{0.5}$ for the dynamical variables e.g. the coordinates of the particles and the conserved quantities e.g. total energy respectively. This error growth is known as Brouwer's Law. Our investigation is based on the implementation of IRK by Hairer *et al.* [45]. They implemented the IRK methods using optimal step-sizes, chosen such that the truncation error was below machine precision. This choice means that the only contribution to the numerical error is due to random round-off errors.

We investigated that the IRK methods having order eight (IRK8) and twelve (IRK12) can be made more efficient by using a higher degree polynomial for the initial estimate of

the stage values (Hairer *et al.* [45] used a linear polynomial). This gave us a significant decrease in the number of iterations and hence in the CPU time, which is worthwhile for long-term simulation of the Solar System. This implementation makes the methods approximately up to 23% more efficient than when implemented by Hairer *et al.* [45]. The numerical experiments show that IRK8 with 6^{th} degree and IRK12 with 2^{nd} degree polynomials are 23% and 19% respectively more efficient for the Jovian Problem. For the Nine Planets Problem, IRK8 and IRK12 with 6^{th} degree polynomials proved to be 15% and 5% more efficient than IRK8 and IRK12 as implemented by Hairer *et al.* [45] respectively. This means that the gain in efficiency using polynomials is problem dependent not method dependent.

We also present comparisons between the IRK methods of Hairer *et al.* [45] and the Störmer methods of Grazier *et al.* [37]. We include the explicit Runge–Kutta Nyström 10-12 pair of Dormand *et al.* [21] to permit a comparison with a method that does not achieve Brouwer's Law.

The provision of continuous extension has been considered using an interpolation polynomial for IRK methods. This is done by fitting a polynomial interpolant to the discrete IRK function evaluations. We implemented the 4^{th} and 6^{th} degree polynomials for IRK8 and IRK12 respectively. Numerical experiments show that the ratio of the error within the interval and mesh points approaches one as the integration proceeds. The continuous extension constructed by Störmer methods using cubic and quintic interpolation is also carried out. We verified the results of Grazier *et al.* [40].

There are still many ideas, we may like to explore. The most obvious is to use a higher degree interpolant. The degree of the interpolant for continuous extensions can be increased using an 8^{th} degree interpolant by implementing a two-step interpolation polynomial. This can be done using the information y_n, y'_n, y''_n at x_n and $y_{n-1}, y'_{n-1}, y''_{n-1}$ and $y_{n-2}, y'_{n-2}, y''_{n-2}$ at x_{n-1} and x_{n-2} respectively. It is expected that these higher degree polynomials will be more efficient than the 6^{th} degree interpolant, used in this study. This two-step interpolant may increase the overhead of the methods.

A

Appendix-A

A.1 Jovian Problem

The Gm for five bodies ordered from Sun, Jupiter, Saturn, Uranus and Neptune are

μ_1 = 0.295912208285591095E-03
μ_2 = 0.282534590952422643E-06
μ_3 = 0.845971518568065874E-07
μ_4 = 0.129202491678196939E-07
μ_5 = 0.152435890078427628E-07

and the initial conditions

	x	y	z
Sun	4.5041709931760E-03	7.629617246855896E-04	2.642173714857008E-04
Jupiter	-5.37970523578697608E+00	-8.30484073974418041E-01	-2.24831631285812891E-01
Saturn	7.89439586897901350E+00	4.59647081929466859E+00	1.55869642252380332E+00
Uranus	-1.82653939237009090E+01	-1.16195110729092122E+00	-2.50107720935801844E-01
Neptune	-1.60550335112138710E+01	-2.39421866167270672E+01	-9.40016532150945853E+00
Sun	-2.686979799291859E-07	5.225296222968518E-06	2.248930945915554E-06
Jupiter	1.09209442155944167E-03	-6.51806804633966371E-03	-2.82076550685720154E-03
Saturn	-3.21747131481691839E-03	4.33585784900737449E-03	1.92866675819078661E-03
Uranus	2.21271749628262749E-04	-3.76242860345373065E-03	-1.65099556049815467E-03
Neptune	2.64285432917179465E-03	-1.49826690091408224E-03	-6.79022140848015384E-04

Table A.1: Rows 1 to 5 list the initial position and rows 6 to 10 the initial velocity.

A.2 Nine Planets Problem

The Gm for ten bodies ordered from Sun to Neptune are

$$\mu_1 = (0.017202098952)^2, \qquad \mu_2 = \mu_1/6023600,$$
$$\mu_3 = \mu_1/408523.5, \qquad \mu_4 = \mu_1/328900.53,$$
$$\mu_5 = \mu_1/3098710, \qquad \mu_6 = \mu_1/1047.355,$$
$$\mu_7 = \mu_1/3498.5, \qquad \mu_8 = \mu_1/22869.0,$$
$$\mu_9 = \mu_1/19314.0, \qquad \mu_{10} = \mu_1/3000000.0$$

and the initial conditions

	x	y	z
Sun	0.9301259103994515E-03	0.2292733100662641E-02	0.9059057664779422E-03
Mercury	0.3448565760800415E+00	0.4790821305397614E-01	-0.1001813144545456E-01
Venus	0.1438953102536455E+00	0.6492977991345496E+00	0.2833883064268579E+00
Earth	-0.1354345700443955E+00	0.8956906559576626E+00	0.3883642504058149E+00
Mars	-0.1368903850273021E+01	0.8454279811185666E+00	0.4247388123779079E+00
Jupiter	0.3350294349606409E+01	-0.3471468715911917E+01	-0.1571243780627322E+01
Saturn	-0.8971574942371711E+01	0.2281974741233523E+01	0.1331244515477938E+01
Uranus	-0.1002073869416921E+01	0.1732580120637246E+02	0.7605730952182388E+01
Neptune	-0.2919365061270080E+02	-0.7716992458897807E+01	-0.2426339472522292E+01
Pluto	-0.2623272065610510E+02	0.2056426815315656E+02	0.1444546303354718E+02
Sun	-0.4559774360194479E-05	-0.3150250493626429E-05	-0.1274328432609927E-05
Mercury	-0.8471091819370054E-02	0.2561145505678817E-01	0.1458557100780699E-01
Venus	-0.1989837205370269E-01	0.3109969215624964E-02	0.2658171477313190E-02
Earth	-0.1732455862288979E-01	-0.2247454982261186E-02	-0.9746354441906539E-03
Mars	-0.7389123605631364E-02	-0.9480508889767826E-02	-0.4152929465094740E-02
Jupiter	0.5581083375222116E-02	0.4959110886728884E-02	0.1991002598306760E-02
Saturn	-0.1862811731356904E-02	-0.4987008831911066E-02	-0.1981531741239860E-02
Uranus	-0.3959813937377914E-02	-0.3790640356065674E-03	-0.1101243197204039E-03
Neptune	0.8161882834578905E-03	-0.2775248510073856E-02	-0.1157390358868530E-02
Pluto	-0.1320448472641354E-02	-0.2623278455987146E-02	-0.4283576834589079E-03

Table A.2: Rows 1 to 10 list the initial position and rows 11 to 20 the initial velocity.

A.3 HRC Problem

The Gm for five bodies ordered from Sun, Jupiter, Saturn, Uranus and Neptune are

$$\begin{aligned}
\mu_1 &= 2.959122082855911025 82E\text{-}4 \\
\mu_2 &= 2.825342103445926254 72E\text{-}7 \\
\mu_3 &= 8.459468504830659292 85E\text{-}8 \\
\mu_4 &= 1.288816238138034888 51E\text{-}8 \\
\mu_5 &= 1.532112481284276189 18E\text{-}8
\end{aligned}$$

and the initial conditions are

	x	y	z
Sun	0.6669198564440767E-02	-0.7235114664408392E-03	-0.1130654423787794E-03
Jupiter	-0.4929481880506559E+01	-0.2310910532399841E+01	0.1197889941614212E+00
Saturn	-0.5559462159881659E+01	0.7217090743352659E+01	0.1008764843911512E+00
Uranus	-0.1051479684851656E+02	-0.1555904864202644E+02	0.7740390484943622E-01
Neptune	0.1636130229890141E+01	0.2982856616501356E+02	-0.6473579962266688E+00
Asteroid	-0.3965267044277659E+01	0.3060320798461592E+00	0.2949122108880113E+00
Sun	-0.1597551822288177E-05	0.7254098157790906E-05	-0.3038348598973975E-07
Jupiter	0.3109433296611612E-02	-0.6477134819096109E-02	-0.4357172559451174E-04
Saturn	-0.4717678753258388E-02	-0.3413503592855709E-02	0.2469252827795303E-03
Uranus	0.3227888778570112E-02	-0.2386568620156909E-02	-0.5061978789868374E-04
Neptune	-0.3152327294479188E-02	0.1931132154044109E-03	0.6952342277721326E-04
Asteroid	-0.1800219023380088E-02	-0.8521337694196810E-02	0.1052106206437703E-03

Table A.3: Rows 1 to 6 list the initial position and rRows 7 to 12 the initial velocity.

A.4 Saturnian Satelites Problem

The Gm for Saturn and its satellites Titan, Hyperion, Iapetus and Rhea are

μ_1 = 8.45945E-8
μ_2 = 2.36777E-4
μ_3 = 0.0000E+0
μ_4 = 3.30000E-6
μ_5 = 4.40000E-6

and the initial conditions for the satellites are

	x	y	z
Titan	-0.0075533871	0.0025250254	-0.0000462204
Hyperion	-0.0006436995	0.0099145485	0.0000357506
Iapetus	0.0219653473	-0.0071369083	0.0062333851
Titan	-0.0010017342	-0.0031443009	0.0000059503
Hyperion	-0.0029182723	0.0000521415	-0.0000356145
Iapetus	0.0006187633	0.0017696165	0.0000439292

Table A.4: Rows 1 to 3 list the initial position and rows 4 to 6 the initial velocity.

B

Appendix-B

B.1 New ERKN 8-10 pairs

The free parameters $c_i, i = 5, 6, \ldots, 11$, for three 8-10 pairs used in section 3.8 of Chapter 3 are given:

f/p	Pair-1	Pair-2	Pair-3
c_5	3.7816356668098677E-01	3.5800754997386142E-01	3.7573634352881985E-01
c_6	2.3340642219668725E-01	2.1676714120826357E-01	2.3145420319518401E-01
c_7	6.9057770903994550E-02	6.2741443345879513E-02	6.7900161894482267E-02
c_8	4.6767082145306405E-01	4.3951960713241267E-01	4.6432886711716126E-01
c_9	6.8384341349308353E-01	6.4658567852808746E-01	6.7535563840305401E-01
c_{10}	6.9075641020317624E-01	6.3334792215508817E-01	6.9035974717019544E-01
c_{11}	8.9994487169379556E-01	8.4908158174866688E-01	8.9119856258216357E-01

B.2 New ERKN 10-12 pairs

The free parameters $c_5, c_6, \ldots, c_{14}, a_{87}, a_{97}, a_{98}, \ a_{1110}, \ a_{1312}$ for three 10-12 pairs used in section 3.8 of Chapter 3 are given:

f/p	Pair-1	Pair-2	Pair-3
c_5	1.6169824256029197E-01	3.6878589554805252E-01	1.3867121921867076E-01
c_6	8.6391183056119564E-02	4.0190789527213705E-01	3.8148564145645636E-01
c_7	6.7774706077147168E-02	6.1835050059732388E-02	5.5099941970174156E-02
c_8	2.1374534011733662E-01	1.9489543428855932E-01	5.5099941970174156E-02
c_9	3.7681169943882814E-01	5.5025819693976585E-01	5.1933709678417905E-01
c_{10}	4.6873712069764029E-01	3.7065168211602806E-01	3.9760816944290622E-01
c_{11}	5.7431288975218664E-01	6.8115508411862535E-01	7.4796391091042980E-01
c_{12}	5.4897073515495354E-01	7.4092473677521020E-01	5.0078078645633151E-01
c_{13}	7.9520877391857303E-01	9.2587683229488060E-01	9.1085406546294456E-01
c_{14}	9.1808540705299513E-01	8.6662294591203137E-01	7.6012586685710715E-01
a_{87}	-1.1181753478716572E-02	1.6154681221438580E-02	-1.3230417684741527E-02
a_{97}	-3.101522980228387E+00	1.7726211852657497E-01	2.8060872097068281E+00
a_{98}	1.3464303908151812E-01	7.3732941401536176E-02	-9.8980026712823591E-02
a_{1110}	1.7495009797235041E-02	4.4121271968179282E-01	-2.8214926501367823E+00
a_{1312}	-1.197827679820685E+00	1.2154507166142844E-01	5.6521143784826409E+00

Bibliography

[1] R. A. Al-Khasawneh, F. Ismael, and M. Suleiman. Embedded diagonally implicit Runge–Kutta Nyström 4(3) pair for solving special second order IVP's. *Appl. Math. Comput.*, 190:1803–1814, 2007.

[2] R. Alexander. Diagonally implicit Runge–Kutta methods for stiff ODE's. *SIAM J. Numer. Anal.*, 14:1006–1021, 1977.

[3] J. H. Applegate, M. R. Douglas, Y. Gursel, G. J. Sussman, and J. Wisdom. The outer Solar System for 200 million years. *Astron. J.*, 92:176–194, 1986.

[4] T. S. Baker, J. R. Dormand, and P. J. Prince. Continuous approximation with embedded Runge–Kutta Nyström methods. *Appl. Numer. Math.*, 29:171–188, 1991.

[5] D. G. Bettis. A Runge–Kutta Nyström algorithm. *Celest. Mech.*, 8:229–233, 1973.

[6] D. Brouwer. On the accumulation of errors in numerical integration. *Astron. J.*, 46:149–153, 1937.

[7] J. C. Butcher. Implicit Runge–Kutta processes. *Math. Comp.*, 18:50–64, 1964.

[8] J. C. Butcher. *Numerical Methods for Ordinary Differential Equations*. Wiley, second edition, 2008.

[9] M. P. Calvo and J. M. Sanz-Serna. The development of variable-step symplectic integrators with applications to the two-body problem. *SIAM J. Sci. Comput.*, 14:936–952, 1993.

[10] M. P. Calvo and J. M. Sanz-Serna. High order symplectic Runge–Kutta Nyström methods. *SIAM J. Sci. Comput.*, 14:936–952, 1993.

[11] B. Cano and B. Archilla. A generalisation to variable stepsizes of Störmer methods for second order differential equations. *Int. J. Appl. Math.*, 19:401–417, 1996.

[12] V. Cerny. A thermodynamic approach to the travelling salesman problem: An efficient simulation. *J. Optim. Theory Appl.*, 45:41–51, 1985.

[13] M. M. Chawla and S. R. Sharma. Intervals of periodicity and absolute stability of explicit Nyström methods for $y'' = f(x, y)$. *BIT*, 21:455–469, 1981.

[14] M. M. Chawla and S. R. Sharma. Absolute stability of explicit Runge–Kutta–Nyström methods for $y'' = f(x, y, y')$. *J. Comput. Appl. Math.*, 10:163–168, 1984.

[15] C. J. Cohen and E. C. Hubbard. Libration of the close approaches of Pluto to Neptune. *Astron. J*, 70:10–13, 1965.

[16] A. Corana, M. Marchesi, C. Martini, and S. Ridella. Minimizing Multimodal Functions of Continuous Variables with the Simulated Annealing Algorithm. *ACM Trans. Math. Software*, 13:262–280, 1987.

[17] C. F. Curtiss and J. O. Hirschfelder. Integration of stiff equations. *Proc. Nat. Acad. Sci.*, 38:235–243, 1952.

[18] R. de Vogelaere. Methods of integration which preserve the contact transformation property of Hamiltonian equations. *Tech. Report No 4, Dept. Mathem., Univ. of Notre Dame, Notre Dame, Ind.*, 4, 1956.

[19] J. R. Dormand. *Numerical Methods for Differential Equations-A Computational Approach*. CRC Press, first edition, 1996.

[20] J. R. Dormand, M. E. A. El-Mikkawy, and P. J. Prince. Families of Runge–Kutta Nyström formulae. *IMA J. Numer. Anal.*, 7:235–250, 1987.

[21] J. R. Dormand, M. E. A. El-Mikkawy, and P. J. Prince. Higher order embedded Runge–Kutta Nyström formulae. *IMA J. Numer. Anal.*, 7:423–430, 1987.

[22] J. R. Dormand and P. J. Prince. A family of embedded Runge–Kutta formulae. *J. Comput. Appl. Math.*, 6:19–26, 1980.

[23] J. R. Dormand and P. J. Prince. Runge–Kutta triples. *Comp. Math. Appl.*, 12A:1007–1017, 1986.

[24] J. R. Dormand and P. J. Prince. New Runge–Kutta algorithms for numerical simulation in dynamical astronomy. *Celest. Mech.*, 18:223–232, 1987.

[25] M. E. A. El-Mikkawy and E. D. Rahmo. A new non-FSAL embedded Runge–Kutta Nyström algorithms of order 6 and 4 in six stages. *Appl. Math. Comput.*, 145:33–43, 2003.

[26] M. E. A. Elmikkawy. Embedded Runge–Kutta Nyström Methods. *Ph.D thesis, University of Teesside, England.*, 1986.

[27] R. England. Error estimates for Runge–Kutta type solutions of ordinary differential equations. *Comput. J.*, 12:166–170, 1969.

[28] W. H. Enright, D. J. Higham, B. Owren, and P. W. Sharp. A survey of the explicit Runge-Kutta method. *Technical Report, 291/94, Department of Computer Science, University of Toronto, Toronto*, 1994.

[29] W. H. Enright, K. R. Jackson, S. P. Norsett, and P. G. Thomsen. Interpolants of Runge-Kutta formulas. *ACM Trans. Math. Software*, 12:193–218, 1986.

[30] E. Fehlberg. Klassische Runge–Kutta formeln fünfter und siebenter ordnung mit schrittweiten-kontrolle. *Computing*, 4:93–106, 1964.

[31] E. Fehlberg. Classical fifth, sixth, seven and eight order Runge–Kutta formulas with stepsize control. *NASA TR R-287*, 1968.

[32] E. Fehlberg. Classical eighth and lower order Runge–Kutta Nyström formulas with stepsize control for special second order differential equations. *NASA TR R-381*, 1972.

[33] K. Feng. On difference schemes and symplectic geometry. *Proceedings of the 1984 Beijing symposium on differential geometry and differential equations, Science Press, Beijing*, 1985.

[34] S. Filippi and J. Graf. Ein Runge–Kutta–Nyström formelpaar der ordnung 11(12) für differentialgleichungen der form $y'' = f(t, y)$. *Computing*, 34:271–282, 1985.

[35] K. R. Grazier. The stability of planetesimal niches in the outer Solar System: A numerical study. *Ph.D thesis, University of California, Los Angeles.*, 1997.

[36] K. R. Grazier, W. I. Newman, D. J. Goldstein, J. M. Hyman, and P. W. Sharp. Brouwer's law: Optimal multistep integrators for celestial mechanics. *Report series 525, Department of Mathematics, University of Auckland*, 2004.

[37] K. R. Grazier, W. I. Newman, J. H. Hyman, J. M. Hyman, and P. W. Sharp. Long simulations of the Solar System: Brouwer's law and chaos. *ANZIAM J.*, 46:C786–C804, 2005.

[38] K. R. Grazier, W. I. Newman, J. M. Hyman, and P. W. Sharp. Achieving Brouwer's law with high order multistep methods. *ANZIAM J.*, 46:C1086–C1103, 2005.

[39] K. R. Grazier, W. I. Newman, W. M. Kaula, and J. M. Hyman. Dynamical evolution of planetesimals in outer Solar System. *ICARUS*, 140:341–352, 1999.

[40] K. R. Grazier, W. I. Newman, and P. W. Sharp. A multirate Störmer algorithm for close encounters. *Submitted to Appl. Math. Comp.*, 2011.

[41] E. Hairer. A One-step method of order 10 for $y'' = f(x, y)$. *IMA J. Numer. Anal.*, 2:83–94, 1982.

[42] E. Hairer. *http://www.unige.ch/ hairer/software.html*, 1995.

[43] E. Hairer. *http://www.unige.ch/ hairer/preprints.html*, 2008.

[44] E. Hairer, C. Lubich, and G. Wanner. *Geometric Numerical Integration. Structure-Preserving Algorithms for Ordinary Differential Equations*. Springer, second edition, 2005.

[45] E. Hairer, R. I. McLachlan, and A. Razakarivony. Achieving Brouwer's law with implicit Runge–Kutta methods. *BIT*, 48:231–243, 2008.

[46] E. Hairer, S. P. Nørsett, and G. Wanner. *Solving Ordinary Differential Equations II: Stiff Problems*. Springer, second edition, 1991.

[47] E. Hairer, S. P. Nørsett, and G. Wanner. *Solving Ordinary Differential Equations I: Nonstiff Problems*. Springer, second edition, 1993.

[48] E. Hairer and G. Wanner. A theory of Nyström methods. *Numer. Math.*, 25:383–400, 1976.

[49] K. Heun. Neue methode zur approximativen integration der differential-gleichungen einer unabhängigen varänderlichen. *Math. Phys.*, 45:23–38, 1900.

[50] N. J. Higham. *Accuracy and Stability of Numerical Algorithms*. Philadelphia: Society for Industrial and Applied Mathematics, first edition, 1996.

[51] M. K. Horn. Development in Higher Order Runge–Kutta Nyström formulas. *Ph.D thesis, University of Texas, Austin.*, 1977.

[52] M. K. Horn. Fourth and fifth order scaled Runge–Kutta algorithms for treating dense output. *SIAM J. Numer. Anal.*, 20:558–568, 1983.

[53] P. J. Van Der Houwen and B. P. Sommeijer. Diagonally implicit Runge–Kutta Nyström methods for oscillatory problems. *SIAM J. Numer. Anal.*, 26:414–429, 1989.

[54] P. J. Van Der Houwen and B. P. Sommeijer. Explicit Runge–Kutta Nyström methods with reduced phase errors for computing oscillating solutions. *SIAM J. Numer. Anal.*, 24:595–617, 1989.

[55] P. J. Van Der Houwen, B. P. Sommeijer, and N. H. Cong. Stability of collocation based Runge–Kutta Nyström methods. *BIT*, 31:469–481, 1989.

[56] T. E. Hull, W. H. Enright, B. M. Fellen, and A. E. Sedgwik. Comparing numerical methods for ordinary differential equations. *SIAM J. Numer. Anal.*, 9:603–637, 1972.

[57] H. Kinoshita and H. Nakai. Motions of the perihelions of Neptune and Pluto. *Celest. Mech.*, 34:203–217, 1984.

[58] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by Simulated Annealing. *Science*, 220:671–680, 1983.

[59] F. T. Krogh. A variable step variable order multistep methods for ordinary differential equations. *Inform. Proc.*, 68:194–199, 1969.

[60] M. W. Kutta. Beitrang zur näherungswenisen integration totaler differentialgleichungen. *Math. Phys.*, 46:435–453, 1901.

[61] J. D. Lambert. *Numerical Methods for Ordinary Differential Equations*. John Willey and Sons, first edition, 1990.

[62] F. M. Lasagni. Canonical Runge–Kutta methods. *Z. Angew. Math. Phys.*, 39:952–953, 1988.

[63] J. Laskar. The chaotic motion of the Solar System-a numerical estimate of the size of the chaotic zones. *ICARUS*, 88:266–291, 1990.

[64] J. Laskar. Large-scale chaos in the Solar System. *Astron. Astrophys.*, 287:9–12, 1994.

[65] J. Laskar, P. Robutel, F. Joutel, M. Gastineau, A. C. M. Correia, and B. Levrard. A long-term numerical solution for the insolation quantities of the Earth. *Astron. Astrophys.*, 428:261–285, 2004.

[66] R. H. Merson. An operation method for the study of integration of study processes. *Proc. Symp. Data processing, weapons study establishment, Australia.*, 1957.

[67] W. E. Milne. A note on the numerical integration of differential equations. *J. Research Nat. Bur. Standards*, 43:537–542, 1949.

[68] E. J. Nyström. Über die numerische Integration von Differentialgleichungen. *Acta Soc. Sci. Fennicae*, 50:1–54, 1925.

[69] D. Okunbor and R. D. Skeel. Explicit canonical methods for Hamiltonian methods. *Math. Comput.*, 59:439–455, 1992.

[70] G. Papageorgiou, I. Th. Famelis, and Ch. Tsitiuras. A p-stable singly diagonally implicit Runge–Kutta Nyström method. *Numer. Algorithms*, 17:345–353, 1998.

[71] B. Paternoster and M. Cafaro. Computation of the interval of stability of Runge–Kutta Nyström methods. *J. Symb. Comput.*, 25:383–394, 1998.

[72] P. J. Prince and J. R. Dormand. Higher order embedded Runge–Kutta formulae. *J. Comput. Appl. Math.*, 7:67–75, 1981.

[73] T. R. Quinn, S. Tremaine, and M. Duncan. A three million year integration of the Earth's orbit. *Astron. J.*, 101:287–305, 1991.

[74] D. L. Richardson and C. F. Walker. Numerical simulation of the nine-body planetary system spanning two million years. *J. Astron. Sci.*, 37:159–182, 1989.

[75] C. Runge. Uber die numerische Auflösung von Differentialgleichungen. *Math. Ann.*, 46:167–178, 1895.

[76] R. D. Ruth. A canonical integration technique. *IEEE Trans. Nucl. Sci.*, 30:2669–2671, 1983.

[77] J. M. Sanz-Serna. Runge–Kutta schemes for Hamiltonian systems. *BIT*, 28:877–883, 1988.

[78] J. M. Sanz-Serna and M. P. Calvo. *Numerical Hamiltonian Problems.* Chapman and Hall, first edition, 1994.

[79] L. F. Shampine. Interpolation of Runge–Kutta methods. *SIAM J. Numer. Anal.*, 22:1014–1027, 1985.

[80] P. W. Sharp. Comparisons of high order Störmer and explicit Runge–Kutta Nyström methods for N-body simulations of the Solar System. *Report series 449, Department of Mathematics, University of Auckland*, 2000.

[81] P. W. Sharp. N-Body Simulations: The performance of some integrators. *ACM Trans. Math. Software*, 32:375–395, 2006.

[82] P. W. Sharp and J. M. Fine. Some Nyström pairs for the general second-order initial value problem. *J. Comput. Appl. Math.*, 42:279–291, 1992.

[83] P. W. Sharp, J. M. Fine, and K. Burrage. Two-stage and three-stage diagonally implicit Runge–Kutta Nyström methods of order three and four. *IMA J. Numer. Anal.*, 10:489–504, 1991.

[84] A. T. Sinclair and D. B. Taylor. Analysis of the orbits of Titan, Hyperion and Iapetus by numerical integration and analytical theories. *Astron. Astrophys.*, 147:241–246, 1985.

[85] B. P. Sommeijer. A note on implicit Runge–Kutta Nyström method. *J. Comput. Appl. Math.*, 19:395–399, 1987.

[86] C. Störmer. Sur les trajectories des coruscules électrisés. *Arch. Sci. Phys. nat. Genéve*, 1907.

[87] Yu. B. Suris. On the canonicity of mappings that can be generated by methods of Runge–Kutta type for integrating systems $x'' = -du/dx$. *Zh. Vychisl. Mat. i Mat. Fiz.*, 2:202–211, 1989.

[88] G. J. Sussman and J. Wisdom. Numerical evidence that the motion of Pluto is chaotic. *Science*, 241:433–437, 1988.

[89] G. J. Sussman and J. Wisdom. Chaotic evolution of the Solar System. *Science*, 257:56–62, 1992.

[90] Ch. Tsitouras and G. Papageorgiou. Runge–Kutta interpolants based on values two successive integration steps. *Computing*, 43:255–266, 1990.

[91] P. J. M. van Laarhoven and E. H. L. Aarts. *Simulated Annealing: Theory and Applications*. D. Reidel Publishing Company, first edition, 1987.

[92] F. Varadi. *http://astrobiology.ucla.edu/ varadi/NBI/ NBI.html*, 1996.

[93] J. H. Verner. Explicit Runge–Kutta methods with estimates of the local truncation error. *SIAM J. Numer. Anal.*, 15:772–790, 1978.

[94] J. H. Verner. Differentiable interpolants for high-order Runge–Kutta methods. *Department of Mathematics and Statistics, Queen's University, Kingston, Canada*, 1990.

[95] H. Yoshida. Construction of higher order symplectic integrators. *Phys. Lett. A*, 150:262–268, 1990.