Comparisons of high order Störmer and explicit Runge-Kutta Nyström methods for N-body simulations of the solar system

P.W. Sharp
Department of Mathematics, University of Auckland
Private Bag 92019, Auckland, NEW ZEALAND.
sharp@scitec.auckland.ac.nz

June 23, 2000

Abstract

We compare the efficiency, stability properties, overhead and storage requirements of fixed-stepsize, high order Störmer and explicit Runge-Kutta Nyström methods for N-body simulations of the solar system. The comparisons of the efficiency are made using realistic problems, one of which requires over 500 million integration steps.

We find high order ERKN methods have better stability properties and smaller overhead than Störmer methods. Our numerical tests suggest ERKN methods are more efficient than Störmer methods for shorter simulations such as one that simulates ten million years of the jovian planets. However, the superior error propagation of the Störmer method means they are more efficient on longer simulations.

Keywords: N-body simulations, Störmer, explicit Runge-Kutta Nyström, high order, comparisons

2000 MSC: Primary - 65L06, secondary - 70F05, 70F10, 85-08

1 Introduction

Simulations of N bodies interacting under Newtonian gravitational forces are used extensively to study the dynamics of the solar system. The studies includes those of the main asteroid belt, Kuiper's belt, the Oort cloud, the long term behaviour of the planets and the origin of the solar system.

These simulations often require the solution of the initial value problem

$$\ddot{y}(t) = f(y(t)), \quad y(t_0) = y_0, \quad \dot{y}(t_0) = \dot{y}_0, \tag{1}$$

where the dot operator denotes differentiation with respect to t and $f : \mathbb{R}^n \mapsto \mathbb{R}^n$.

High order methods are used to solve (1) when the truncation error on each step must be small. Often the stepsize is chosen so that the truncation error is at or below the limit of double precision accuracy (about 16 significant figures). These simulations can have n large and can require copious CPU time. For example, the simulation of 10^9 years of the outer solar system describe in [6] had $n = 600,030$, required 9×10^{10} integration steps and took the equivalent of three months of CPU time on a single workstation.

Some simulations use more than one type of particle, with massive and massless particles being the most common types. Massive particles are used to model large bodies in the solar system such as the Sun and the planets and massless particles to model small bodies such as planetesimals, small asteroids and comets. Massive particles attract both types of particles, massless particles do not attract other particles. In many simulations the number of massive particles is small, frequently no more than 10; large values of n usually come from having a large number of massless particles.

For a simulation of N_M massive particles and N_m massless particles the cost of evaluating f using direct summation is proportional to

$$(N_M - 1)N_M + N_M N_m. \tag{2}$$

The first term represents the interactions between massive particles and the second term the interactions between massive and massless particles. For large N_m/N_M, the cost of evaluating f varies linearly with n and the cost of evaluating each component of f is then small if N_M is small.

Pride of place among high order methods are fixed-stepsize, fixed-order Störmer methods. Grazier, Newman, Kaula and Hyman [6] state: *In the mid-1960s through the early 1990s, Störmer schemes became the standard integration methods for celestial mechanics. Concurrently and subsequently, many other astronomical and planetary dynamicists have employed this methodology.*

Another possible class of high order methods is non-symplectic explicit Runge-Kutta Nyström (ERKN) methods. These methods are considerably more efficient than explicit Runge-Kutta methods applied to the equivalent first order problem. This raises the interesting question of whether ERKN methods are more efficient than Störmer methods. Given the large amount of CPU time required for some simulations, even a small gain in efficiency is worth having.

We compare the efficiency, stability properties, overhead and storage requirements of high order fixed-stepsize ERKN and Störmer methods. We measure efficiency by the amount of work required to achieve a prescribed accuracy. Two obvious measures of work are the number of f evaluations and the amount of CPU time. The number of evaluations has the advantage of being machine independent. However, if the cost of evaluating each component of f is small, the overhead of the integration method will make a significant contribution to the total CPU time. It could then be argued that CPU time is a fairer measure of work for these problems. Both measures of work have their merits and we use both in our comparisons.

We define the methods in §2 and compare their stability properties, overhead and storage requirements in §3. We also summarise techniques for controlling the growth of round-off error. In §4 we present our numerical comparisons of the methods. We end in §5 with a summary of our work and a discussion of its possible implications.

2 Methods

Each method forms a numerical approximation to $y(t_i)$ and $\dot{y}(t_i)$ at $t_i = t_0 + ih$, $i = 1, 2, \ldots,$. The approximations are denoted by y_i and $\dot{y}_i$ respectively.

An order $k+1$ Störmer method when written in the function form, is defined as

$$
\begin{aligned}
y_i &= 2y_{i-1} - y_{i-2} + h^2 \sum_{l=0}^{k} \alpha_l \ddot{y}_{i-1-l}, \\
\dot{y}_i &= h^{-1}(y_{i-1} - y_{i-2} + h^2 \sum_{l=0}^{k} \beta_l \ddot{y}_{i-1-l}),
\end{aligned}
\tag{3}
$$

where $\ddot{y}_l = f(y_l)$. The starting values $y_1, y_2, \ldots, y_k$ are often calculated using a one-step method. We use the extrapolation method described on pages 271 and 273 of [7].

The velocity is not required in some simulations. The update formula for $\dot{y}$ can then be omitted, reducing the overhead. We return to this point later.

Numerical experiments (see, for example, [5]) suggest the order 13 Störmer method is often more efficient than Störmer methods of other orders and we use it in our comparisons. To illustrate the dependence on order, we also use the Störmer methods of orders 9 and 11.

An s-stage ERKN method forms the approximations y_i and $\dot{y}_i$ using

$$
\begin{aligned}
y_i &= y_{i-1} + h\dot{y}_{i-1} + h^2 \sum_{j=1}^{s} b_j f_j, \\
\dot{y}_i &= \dot{y}_{i-1} + h \sum_{j=1}^{s} b'_j f_j,
\end{aligned}
\tag{4}
$$

$$
f_1 = f(y_{i-1}), \quad f_j = f(y_{i-1} + c_j h \dot{y}_j + \sum_{l=1}^{j-1} a_{jl} f_l), \quad j = 2, \ldots, s.
$$

The coefficients b_j, c_j and a_{jk} are chosen so the method has the required order and desirable properties such as small error coefficients. Unlike Störmer methods, $\dot{y}_i$ must be calculated on each step.

The numerical experiments of Dormand, El-Mikkawy and Prince [2] suggest the 17-stage, order 12 ERKN method in [2] is often the most efficient among high order ERKN methods and we use it in our comparisons. To illustrate the dependence on the order, we also use the order 8 ERKN method in [2].

We denote the order 8 and order 12 ERKN methods by the acronyms ERKN8 and ERKN12 respectively, and the Störmer method of order q by Sq.

3 Non-numerical comparisons

3.1 Stability

The equation of motion for orbital dynamics is related to the equation for an harmonic oscillator

$$\ddot{y} = -\omega^2 y, \quad \omega > 0. \tag{5}$$

When the Störmer method (3) is applied to (5), the resulting difference equation is

$$\sum_{j=0}^{k+1} \gamma_j y_{i-j} = 0, \tag{6}$$

where $\gamma_0 = 1$, $\gamma_1 = -2 + \bar{h}^2\alpha_0$, $\gamma_2 = 1 + \bar{h}^2\alpha_1$, $\gamma_j = \bar{h}^2\alpha_{j-1}$, $j = 3, \ldots, k+1$ and $\bar{h} = h\omega$.

The stability polynomial $P(z; \bar{h})$ is defined as $\gamma_0 r^{k+1} + \cdots + \gamma_{k+1}$. When $\bar{h} = 0$, P has two principal roots at $z = 1$ and $k-1$ extraneous roots at $z = 0$. The interval of absolute stability is defined (see, for example, [4]) as the interval $[0, \bar{h}_s]$ such that all extraneous roots of $P(z; \bar{h})$ are no greater than one in magnitude.

The interval of stability for S9, S11 and S13 is $[0, 0.334]$, $[0, 0.175]$ and $[0, 0.090]$ respectively, where the right end point is given to three decimal places.

When an s-stage ERKN method is applied to (5), we obtain

$$Y_i = R(z)Y_{i-1},$$

where

$$Y_i = \begin{bmatrix} y_i \\ h\dot{y}_i \end{bmatrix}, \quad z = -h^2\omega^2,$$

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

$$b = [b_1, \ldots, b_s]^T, \quad b' = [b'_1, \ldots, b'_s]^T, \quad c = [c_1, \ldots, c_s]^T, \quad A = \{a_{ij}\}_{i,j=1}^s, \quad e = [1, \ldots, 1]^T,$$

with $c_1 = 0$, $a_{ij} = 0, j \geq i$.

If $\rho(R)$ denotes the spectral radius of $R(z)$, the interval $[0, \sqrt{-z_0}]$, $z_0 < 0$, on which $\rho(R) \leq 1$ is called the interval of stability.

This interval is $[0, 3.292]$ and $[0, 8.326]$ for ERKN8 and ERKN12 respectively. To make a fair comparison with the Störmer methods, we need to scale the intervals by the amount of work done on each step. One possible measure is the number of derivative evaluations. The intervals for the ERKN methods become $[0, 0.411]$ and $[0, 0.490]$ which compare favourably with the intervals for S9, S11 and S13.

Another test equation used to investigate the stability is

$$\dot{y} = \lambda y, \quad \mathrm{Re}(\lambda) \leq 0. \tag{7}$$

The interval of absolute stability becomes a region of absolute stability.

Figure 1 gives the scaled stability regions for the five methods. The stability regions for the Störmer methods decrease in size with order and the region for ERKN8 is larger than that for ERKN12. The regions for ERKN8 and ERKN12 are larger than those for the Störmer methods.

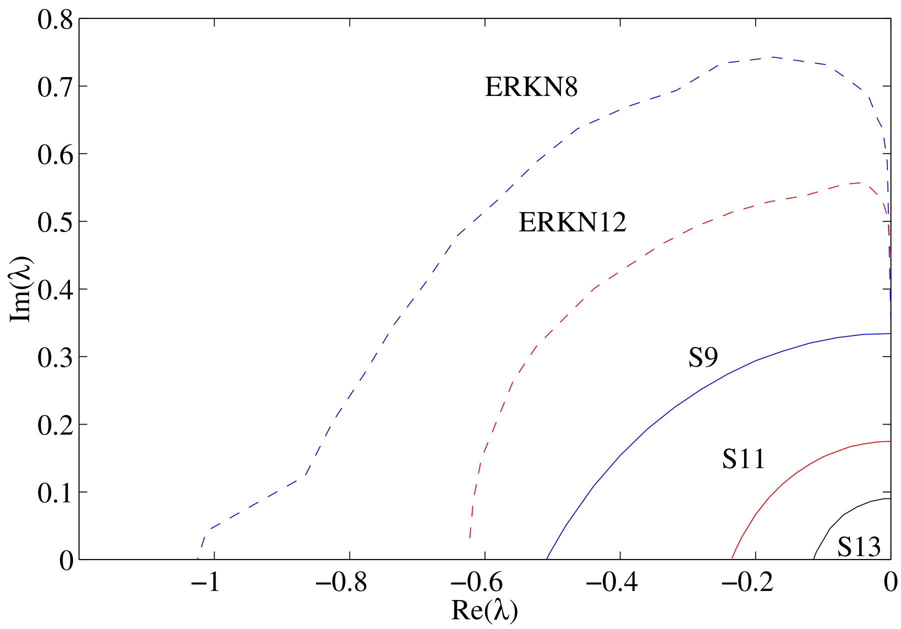

Figure 1: The scaled stability regions of S9, S11, S13, ERKN8 and ERKN12 for (7).

3.2 Round-off error

The growth in round-off error for Störmer methods can be limited by using the backward difference form of the methods, and by implementing these differences in the summed form and adding them from the highest to lowest order. These techniques for limiting the round-off error do not increase the overhead. The propagation of the round-off error for Störmer methods is discussed in detail in [5].

The backward difference form of the formulae for y_i and $\dot{y}_i$ are

$$\begin{aligned} y_i &= 2y_{i-1} - y_{i-2} + h^2 \sum_{l=0}^{k} \gamma_l \nabla_l \ddot{y}_{i-1}, \\ \dot{y}_i &= h^{-1}(y_{i-1} - y_{i-2} + h^2 \sum_{l=0}^{k} \delta_i \nabla_l \ddot{y}_{i-1}), \end{aligned}$$

where $\nabla_0 \ddot{y}_{i-1} = \ddot{y}_{i-1}$ and $\nabla_j \ddot{y}_{i-1} = \nabla_{j-1}\ddot{y}_{i-1} - \nabla_{j-1}\ddot{y}_{i-2}$. The coefficients γ_l and ∇_l can be calculated from the generating functions

$$\sum_{l=0}^{\infty} \gamma_l x^l = \left[\frac{x}{\log_e(1-x)}\right]^2 \frac{1}{1-x}, \quad \sum_{l=0}^{\infty} \delta_l x^l = \left[\frac{x^2 - x - \log_e(1-x)}{(\log_e(1-x))^2(1-x)}\right]. \tag{8}$$

In the summed form, the formula for y_i becomes

$$y_i = y_{i-1} + \phi_i, \quad \phi_i = \phi_{i-1} + h^2 \sum_{l=0}^{k} \gamma_l \nabla_l \ddot{y}_{i-1-l}, \quad \phi_1 = y_1 - y_0. \tag{9}$$

There has been little work on developing techniques to limit the growth of round-off error in ERKN methods. However, the form of an ERKN method has much in common with an explicit Runge-Kutta method and the techniques of Gill [3] and Møller [9], [10] are relevant. Kouya [8] compared these two techniques for a selection of explicit Runge-Kutta methods and concluded that Møller's technique was just as effective as Gill's technique. We chose Møller's technique because it has less overhead.

In Møller's technique, the update formula for y_i is replaced by the sequence

$$\tau = \sum_{j=1}^{s} b_j f_j - \epsilon, \quad y_t = y_{i-1} + h\dot{y}_{i-1} + \tau, \quad \epsilon = (y_t - y_{i-1} - h\dot{y}_{i-1}) - \tau, \quad y_i = y_t,$$

where $\epsilon = 0$ at the start of an integration. The update formula for $\dot{y}$ is modified in a similar way.

3.3 Overhead

For an order k Störmer method in which the position and velocity are calculated, $(2k+3)n$ multiplications and $(2k+3)n$ additions are required to form y_i and $\dot{y}_i$. Another $(k-1)n$ additions are required to update the differences. Hence the overhead for one step is $(2k+3)n$ multiplications and $(3k+2)n$ additions. This overhead is also the overhead per derivative evaluation since one evaluation is performed per step.

The overhead for an ERKN method is not easily expressed as a function of the order because more than one method of the same order is possible and these methods may differ in the number of stages and non-zero coefficients. ERKN8 requires $60n$ multiplications and $51n$ additions per step. The overhead per derivative evaluation is $7.5n$ multiplications and $6.4n$ additions. The corresponding figures for ERKN12 are $6.5n$ multiplications and $5.7n$ additions.

The above counts per evaluation show ERKN8 requires fewer than half the number of multiplications of S8 and ERKN12 requires one quarter the number of multiplications of S12. The differences are greater for additions.

If the velocity is not calculated when using a Störmer method, the number of multiplication is reduced by one half and the number of additions by one third. ERKN8 then requires one third fewer multiplications than S8 and ERKN12 requires less than one half the number of multiplications of S12.

3.4 Storage

In simulations of a large number of bodies, the amount of storage required by an integration method can be a limiting factor.

An order k Störmer method requires kn locations to store the divided differences and n locations for each of y_{i-1} and $\dot{y}_{i-1}$ (which are overwritten by y_i and $\dot{y}_i$ respectively), to give a total of $(k+2)n$ locations. Since in general divided differences are independent of one another, the only way to reduce the storage requirements of a Störmer method is to reduce its order. This often leads to more evaluations of f being required to achieve the same accuracy.

If no storage reduction techniques are used, ERKN12 requires $17n$ locations for f_j, $j = 1, \dots, 17$, together with n locations for the argument of f_j, and n locations for each of y_{i-1} and $\dot{y}_{i-1}$, to give a total of $20n$ locations. This total is easily reduced to $18n$ because $a_{i2} = a_{i3} = 0$, $i = 6, \dots, 17$ which means f_2 and f_3 are not needed to evaluate $f_6, \dots, f_{17}$. A further $6n$ locations can be saved by re-ordering the calculations of the arguments to f_j, $j > 11$. The overhead changes marginally. This last technique for reducing storage requirements is illustrated in the integrator DOPRI8 on page 437 of [7]. Reductions in the storage requirements for ERKN8 can also be made.

We observe from the above arguments that the storage requirements are similar for the Störmer and ERKN methods we are interested in.

4 Numerical comparisons

The numerical comparisons were made on a single processor of a SGI Origin 2000 computer. The programs were written in Fortran 90 and compiled using the compiler options `-64`, `-mips4`, `-r10000` and `-O3`. Some integrations were performed in double precision (about 16 significant figures), others in quadruple precision (about 32 significant figures). We used quadruple precision to help us assess the effects of round-off error for simulations in double precision.

The Störmer methods were implemented in the backward difference summed form and the differences were added from the highest to lowest order. The ERKN methods were implemented using the round-off control of §3.2.

As noted in §1, the stepsize is often chosen so that the truncation error on each step is at or below the limit of double precision accuracy. We used such stepsizes in most of our simulations.

The energy is a conserved quantity in all of our simulations. At time t, the energy $E(t)$ is defined as

$$E(t) = \frac{1}{2}\sum_{j=1}^{N} m_j v_j^2 - \sum_{i=1}^{N-1}\sum_{j=i+1}^{N} \frac{Gm_i m_j}{r_{ij}},$$

where N is the number of bodies in the simulation, m_j is the mass of the j^{th} body, v_j is its speed at time t, G is the gravitational constant and r_{ij} is the distance between the i^{th} and j^{th} particle.

The relative error in the energy is defined as

$$\frac{GE(t) - GE(0)}{GE(0)},$$

where we have multiplied the energy by G because the product Gm_j of celestial bodies is usually known far more accurately than either G or m_j.

The energy requires little CPU time to calculate. In addition, since the true value is known, the true error in the energy can be calculated. However, the error in the energy is usually less than that in the solution. This gives a false impression of the accuracy of a solution. Hence, in the results below we have (where appropriate) given the estimated error in the solution.

In some graphs, we have used the scaled stepsize as the abscissa. This stepsize is the stepsize used in an integration divided by the number of derivative evaluations per step. For a Störmer method, the stepsize and the scaled stepsize are the same; for ERKN8 and ERKN12 the scaled stepsize is 1/8 and 1/17 of the stepsize respectively.

4.1 Kepler's problem

The first set of comparisons is for Kepler's problem with eccentricities less than one. The solution of this problem is used in a number of important applications including the orbit determination of new asteroids, and the calculation of the orbit for the two primary bodies in a restricted three-body problem.

The equations of motion are

$$\ddot{y}_1 = -\frac{y_1}{r^3}, \quad \ddot{y}_2 = -\frac{y_2}{r^3}, \quad r = (y_1^2 + y_2^2)^{1/2},$$

where the subscripts on y here refer to the equation number and not the step number. The initial conditions are $y_1(0) = 1 - e$, $\dot{y}_1(0) = 0$, $y_2(0) = 0$ and $\dot{y}_2(0) = \sqrt{((1+e)(1-e)^{-1}}$ where e is the eccentricity ($0 \leq e < 1$). The solution is $y_1(t) = \cos E - e$, $y_2(t) = \sqrt{1.0 - e^2}\sin E$, $\dot{y}_1(t) = -\sin E(1 - e\cos E)^{-1}$, and $\dot{y}_2(t) = \sqrt{(1-e^2)}\cos E(1-e\cos E)^{-1}$, where the eccentric anomaly E satisfies Kepler's equation $E - e\sin(E) - t = 0$. The solution is periodic with period 2π.

Figure 2 shows the maximum error in $y_1(10000)$ as a function of the scaled stepsize for $e = 0$ and 0.5 (the osculating eccentricity of the planets in the solar system are less than 0.26). The integrations were performed in quadruple precision. For $e = 0$, the methods rank in efficiency according to their order, except near limiting precision where it appears the accumulated round-off error alters the relative efficiency. This ranking holds for $e = 0.5$ except there is little difference between the efficiency of ERKN12 and S11, and the accumulated round-off error does not appear significant (for the stepsizes we used). The relative efficiency when y_2, $\dot{y}_1$ or $\dot{y}_2$ are used to make the comparisons is almost identical to that for y_1.

Figure 3 shows the relative error in the energy at $t = 10000$. The relative efficiency of the methods is similar to that in Figure 2.

Using the error at one point to compare methods can be misleading if the error oscillates with t. Figure 4 shows the absolute error in y_1 and the relative error in the energy as a function of t for $e = 0.5$. The scaled stepsize was 0.00125 for each method. The methods rank in efficiency as in Figure 3 with the error growing as t^2 for large t. More interestingly, the error in y_1 is oscillatory with a period of about 185 times the orbital period of the two bodies. The reason for this oscillatory behaviour is unknown to us.

4.2 Jovian problem

The next set of comparisons is simulations of the Sun and the four jovian planets (Jupiter, Saturn, Uranus and Neptune). Simulations of these bodies are of tremendous importance because these bodies play a fundamental role in the solar system. For example, Jupiter, by sweeping up debris which may have bombarded the Earth, was crucial to the evolution of life on Earth.

Let $\mathbf{r}_i$, $i = 1, \ldots, 5$, be the position in three dimensions of the i^{th} body. Then

$$\ddot{\mathbf{r}}_i = \sum_{j=1,\ j\neq i}^{5} \frac{Gm_j(\mathbf{r}_j - \mathbf{r}_i)}{r_{ij}^3}, \quad r_{ij}^2 = (\mathbf{r}_j - \mathbf{r}_i)\cdot(\mathbf{r}_j - \mathbf{r}_i).$$

We used the initial conditions in Table 1. Rows 1, 3, 5, 7, 9 give the position Sun, Jupiter, Saturn, Uranus and Neptune respectively; rows 2, 4, 6, 8, 10 give the velocity. The initial conditions are those supplied with the integrator `NBI` [12]. The units of distance and time are one astronomical unit and one Julian day respectively.

4.514411871435666407e-003	7.228284115206586734e-004	2.465910049256798627e-004
-2.836944634081315163e-007	5.181194408646325544e-006	2.230658834062126348e-006
-5.389682454460906133e+000	-7.702654951861659303e-001	-1.986643116590752201e-001
1.005345256992409818e-003	-6.529842519168941664e-003	-2.825878753242960953e-003
7.952776853025736086e+000	4.507882218400655368e+000	1.520195525318333889e+000
-3.159466250401293011e-003	4.371463427837262235e-003	1.944139516913710376e-003
-1.827823658635314753e+001	-9.576457288148205643e-001	-1.613219039727103541e-001
1.710831056480681724e-004	-3.764670468281590004e-003	-1.651967861025700013e-003
-1.636719135877088833e+001	-2.376089672537307634e+001	-9.321386617949729010e+000
2.622524276428921378e-003	-1.527747312385890404e-003	-6.918319756218280486e-004

Table 1: The initial conditions for the Jovian problem.

The product Gm_j for the five bodies is

$$\begin{aligned} Gm_1 &= 2.95912208285591102582e-4, & Gm_2 &= 2.82534210344592625472e-7, \\ Gm_3 &= 8.45946850483065929285e-8, & Gm_4 &= 1.28881623813803488851e-8, \\ Gm_5 &= 1.53211248128427618918e-8, \end{aligned}$$

where the mass is in solar masses. These values are supplied with the integrator `DE118i` [11].

Figure 5 gives the relative error in the energy for a simulation in quadruple precision of one million years. The stepsize for the Störmer methods was four days, a value very similar to that used in [6]. The stepsize for ERKN8 and ERKN12 was 32 and 68 days respectively which meant the scaled stepsize was the same for all five methods. The methods rank in efficiency by order except for S11 and ERKN12 which are of similar efficiency. We examined the error in more detail and found it grew linearly with t for all methods (see, for example, [1]).

We repeated the previous simulation in double precision and found there was a marked difference between the Störmer and ERKN methods. The relative error in the energy for the Störmer methods appeared to grow stochastically $t^{1/2}$, as was found in [5] and [6]. In contrast, the growth was linear for the ERKN methods.

We examined the apparent stochastic behaviour for the Störmer methods by using a sampling technique similar to that in [5]. For each method we performed sixteen simulations. The initial conditions for the first simulation were those given in Table 1. The initial conditions for the i^{th} simulation, $i > 1$, were the position and velocity of the bodies at $t = (i-1)h$. The positions and velocities were obtained by taking $i-1$ steps of size h with ERKN12. After some experimentation, we chose $h = 28$. This value ensured the truncation error on each step was well below the limit of double precision accuracy.

For each method, we used the sixteen simulations to calculate the root mean square (rms) in the relative error of the energy as a function of t. To gain a clearer understanding of the behaviour of the error, we extended the integration from one million years to six million years, six million being a compromise between increased insight and availability of computer resources.

The results are displayed in Figure 6 with the errors for S9, S11 and S13 represented by $+$, $\bigcirc$ and $\diamondsuit$ respectively. Out to approximately 10,000 years (912,500 integration steps), the rms error is almost constant. For larger t, the rms error increases steadily. The lines through the data points are lines of linear regression for $t \geq 10^4$ years. The slopes of the lines are 0.48, 0.41 and 0.44 for S9, S11 and S13 respectively, in good agreement with the theoretical value of 0.50 for stochastic growth (see, for example, [5]).

Figure 7 is Figure 6 with the relative error in the energy for ERKN8 and ERKN12 added. The error for each ERKN method was calculated from one integration. The error for ERKN8 is less than that for ERKN12, suggesting ERKN8 has better round-off error propagation properties than ERKN12. Another interesting observation is that although the error for ERKN8 grows linearly with t as against the square root of t for S9, S11 and S13, ERKN8 is more accurate than the Störmer methods for the six million years. An extrapolation of the data in Figure 7 suggests the Störmer methods do not become more efficient than ERKN8 until at least ten million years.

Figure 8 gives the error (rms error for S9, S11 and S13) in the position of Jupiter. The errors for the Störmer methods are similar and grow approximately as $t^{3/2}$, agreeing with the results in [5] and [6]. The error for ERKN8 and ERKN12 grows approximately as t^2. An extrapolation of the data in Figure 8 suggests ERKN8 will be more efficient than the Störmer methods until approximately 20 million years.

4.3 Nine Planets problem

The third set of comparisons is simulations of the Sun and the nine planets. The equations of motion are the same as for the Jovian problem, except there is ten bodies instead of five. The initial conditions and Gm_j are listed in the Appendix.

The orbital period of the terrestrial planets (Mercury, Venus, Earth and Mars) are shorter than for the jovian planets. This means the scaled stepsize must be smaller than for the Jovian problem. A comparison of the period for Mercury (88 days) and Jupiter (4333 days) suggests the stepsize should be 50 times smaller. Using this value as a guide, we did some numerical testing and found a stepsize of 0.1 days (40 times smaller than for the Jovian problem) was satisfactory.

Figure 9 shows the relative error in the energy for a simulation in double precision of 50,000 years (equivalent to approximately two million years for the Jovian problem). The error for S9, S11 and S13 is the rms error and the error for ERKN8 and ERKN12 is from one integration. The error for S9, S11 and S13 varies little over the 50,000 years and is similar for the three methods. The error for ERKN8 and ERKN12 is smaller than for the Störmer methods, but grows linearly with time.

4.4 CPU time

Table 2 gives the average CPU time per step for the Jovian and Nine Planets problems. The times are in seconds. A single estimate of the time was found by measuring the time for a large number of integration steps and then dividing by the number of steps. Five estimates were found for each method and problem, and averaged to give the values in Table 2. The large number of steps meant we could neglect the extra time required by the extrapolation method at the start of an integration with a Störmer method.

Problem	S9	S11	S13	ERKN8	ERKN12
Jovian	2.38×10^{-5}	2.65×10^{-5}	2.84×10^{-5}	5.83×10^{-5}	1.30×10^{-4}
Nine Planets	5.83×10^{-5}	6.39×10^{-5}	6.83×10^{-5}	1.96×10^{-4}	4.25×10^{-4}

Table 2: The average CPU time (in seconds) per step.

Since the overhead for Störmer methods increases with order and they perform the same number of derivative evaluations per step, we expect the time per step for Störmer methods to increase with order. The times in Table 2 support this expectation.

The operation counts of §3.3 show the overhead per derivative evaluation for ERKN8 and ERKN12 are similar. This suggests the ratio of the time for ERKN12 and ERKN8 should be close to $17/8 (= 2.125)$, the ratio of the number of stages. In addition, the evaluation of f for the Nine Planets problem requires approximately four times the CPU time as the evaluation for the Jovian problem. This implies the ratio of times for the Nine Planet problem should be closer to 17/8 than the ratio for the Jovian problem. The ratios are 2.22 and 2.17, in very good agreement with expectations.

The Störmer methods require more time than the ERKN methods than predicted by the ratio of the number of derivative evaluations per step. This still holds if the velocity is not calculated when using Störmer methods, although the difference is not as large.

4.5 Massless

The last set of simulations is of the Sun, the jovian planets and N_m massless particles, where several values of N_m in the range 100 to 1000 were used. Since $N_M = 5$, the cost of evaluating f is (from (2)) proportional to $20 + 5N_m$. For the values of N_m we are interested in, the cost is effectively linear in N_m.

The initial conditions and Gm_j were those for the Jovian problem. The initial conditions of the massless particles were specified as in [6]. The inclination of the orbits were normally distributed about the ecliptic, the semi major axes were normally distributed between the orbits of Jupiter and Saturn, the eccentricity was chosen from a negative exponential distribution, and the remaining three orbital elements were randomly and uniformly distributed on the interval $[0, 2\pi]$.

The results of these comparisons supported the results of the previous comparisons.

5 Discussion

We compared the Störmer methods of orders 9, 11 and 13 with two non-symplectic explicit Runge-Kutta Nyström (ERKN) methods, one of order 8 and one of order 12. The two ERKN methods were selected as being representative of ERKN methods of high order. Our main aim was to assess the relative efficiency of high order ERKN and Störmer methods on N-body simulations of the solar system. We also compared the size of the stability intervals and regions, the overhead and the storage requirements.

There are clear differences between the ERKN and Störmer methods. The scaled stability intervals for the harmonic oscillator test equation $\ddot{y} = -\omega^2 y$, where the scaling is by the number of derivative evaluations, are larger for the ERKN methods. This advantage also holds for the more general test equation $\ddot{y} = \lambda^2 y$, $\mathrm{Re}(\lambda) \leq 0$.

The overhead per derivative evaluation is smaller for the ERKN methods than for the Störmer methods. If the position and velocity are calculated on each step of the Störmer method, the overhead of the ERKN methods is no more than half that of the Störmer methods. If just the position is calculated, the overhead of the ERKN methods is at least one third smaller than for the Störmer methods. Hence when the same scaled stepsize is used for the Störmer and ERKN methods, the ERKN methods require less CPU time to complete the simulation.

Another clear difference is the propagation of error for simulations in double precision with the truncation error at or below the limit of double precision accuracy. This was well illustrated by the error in the energy. If a Störmer method is implemented in the backward difference summed form with the differences added from the highest to lowest order, the error grows approximately as the square root of time. For an ERKN method implemented with Møller's technique, the error grows linearly with time.

We performed some simulations in quadruple precision. This permitted truncation errors at the limit of double precision accuracy without round-off error being significant and provided valuable information on the efficiency. We found the methods could usually be ranked in efficiency according to their order, with the method of highest order being the most efficient. One exception was that the order 11 Störmer method and the order 12 ERKN method were often of similar efficiency.

We found ERKN methods could be more efficient for the first part of long simulations when the calculations were done in double precision (as most simulations would be). For example, in a simulation of the Sun and jovian planets, the order 8 ERKN method was more efficient than the Störmer methods for the first six million years. An extrapolation of our data suggests the ERKN method would be more efficient until at least ten million years.

The above conclusions on the relative efficiency were made using the number of derivative evaluations as the measure of work. If CPU time is used, the efficiency of the ERKN methods relative to the Störmer methods increases. As noted above, this increase will depend on whether the velocity is calculated on each step when using the Störmer methods. There is also the caveat that the increase depends on the implementation of the methods and is machine dependent.

Our overall conclusion is that when simulations are done in double precision, high order ERKN methods are potentially more efficient than high order Störmer methods for short simulations, but are less efficient for long simulations.

Acknowledgements

The author thanks Dr. F. Krogh for his helpful discussions on the stability of Störmer methods and for making comments on an early draft of the paper.

References

[1] M.P. Calvo, J.M. Sanz-Serna, *The development of variable-step symplectic integrators with applications to the two-body problem*, SIAM J. Sci. Comput. **14** (1993), 4, 953-970.

[2] J.R. Dormand, M.E.A. El-Mikkawy, P.J. Prince, *High-order embedded RKN formulae*, IMA J. Num. Analysis **7** (1987), 423-430.

[3] S. Gill, *A process for the step-by-step integration of differential equations in an automatic digital computing machine*, Proc. Camb. Phil. Soc., **47** (1951), 96-108.

[4] D.J. Goldstein, *The near optimality of Störmer method for long time integration of* $y'' = f(x, y)$, Ph.D. thesis, University of California, Los Angeles, 1996.

[5] 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.

[6] K. Grazier, W.I. Newman, W.M. Kaula, J.M. Hyman, *Dynamical evolution of planetesimals in the outer solar system: I. the Jupiter/Saturn zone*, Icarus **140** (1999), 341-352.

[7] E. Hairer, S.P. Nørsett, G. Wanner, *Solving ordinary differential equations I: nonstiff problems*, Springer-Verlag, 1987.

[8] T. Kouya, *A comparison between the Møller and the Gill methods for explicit Runge-Kutta process*, Annals of Num. Math. **1** (1994), 411-422.

[9] O. Møller, *Quasi double-precision in floating point addition*, BIT **5** (1965), 37-50.

[10] O. Møller, *Note on quasi double-precision*, BIT **5** (1965), 251-255.

[11] S. Moshier, *DE118i*, `http://people.ne.mediaone.net/moshier/index.html`

[12] F. Varadi, *NBI*, `http://www.astrobiology.ucla.edu/~varadi/NBI/NBI.html`

Appendix

Table 3 contains the initial conditions for the Nine Planets problem. The odd number rows are the initial positions and the even numbered rows are the initial velocities. The bodies are ordered Sun, Mercury, Venus, Earth, Mars, Jupiter, Saturn, Uranus, Neptune and Pluto.

The values of Gm_j for the Nine Planets problem are given below. The bodies are ordered as for Table 3. The values of Gm_j for the Sun and the jovian planets are the same as for the Jovian problem.

$$\begin{array}{rclrcl}
Gm_1 &=& 2.95912208285591102582e-4, & Gm_2 &=& 4.91254745145081175785e-11, \\
Gm_3 &=& 7.24345620963276523095e-10, & Gm_4 &=& 8.88769273403302327042e-10, \\
Gm_5 &=& 9.54952894222405763492e-11, & Gm_6 &=& 2.82534210344592625472e-7, \\
Gm_7 &=& 8.45946850483065929285e-8, & Gm_8 &=& 1.28881623813803488851e-8, \\
Gm_9 &=& 1.53211248128427618918e-8, & Gm_{10} &=& 2.27624775186369921644e-12.
\end{array}$$

The initial conditions and values of Gm_j are supplied with the integrator `DE118i` [11].

4.5144118714356666407e-003	7.2282841152065867346e-004	2.4659100492567986271e-004
-2.8369446340813151639e-007	5.1811944086463255444e-006	2.2306588340621263489e-006
3.6030663368975339466e-001	-9.4812876741771684223e-002	-8.7466840117233140436e-002
3.7085069798210382186e-003	2.4854958767430945324e-002	1.2929109014677844626e-002
6.0786466491731583464e-001	-3.5518362463675619232e-001	-1.9824142909855515515e-001
1.1156645711264016669e-002	1.5494075513638794325e-002	6.2773904546696609267e-003
0.1082176318288926413e+000	-0.9270869831537622570e+000	-0.4020803162138249869e+000
0.1682597583356446303e-001	0.1562319686603147364e-002	0.6775367895532445041e-003
-1.2796408611369531836e-001	-1.3262618005333617013e+000	-6.0530808652523961512e-001
1.4481919298277924969e-002	8.0528538390447499843e-005	-3.5188931029397090065e-004
-5.3896824544609061333e+000	-7.7026549518616593034e-001	-1.9866431165907522014e-001
1.0053452569924098185e-003	-6.5298425191689416643e-003	-2.8258787532429609536e-003
7.9527768530257360864e+000	4.5078822184006553686e+000	1.5201955253183338898e+000
-3.1594662504012930114e-003	4.3714634278372622354e-003	1.9441395169137103763e-003
-1.8278236586353147533e+001	-9.5764572881482056433e-001	-1.6132190397271035415e-001
1.7108310564806817248e-004	-3.7646704682815900043e-003	-1.6519678610257000136e-003
-1.6367191358770888335e+001	-2.3760896725373076342e+001	-9.3213866179497290101e+000
2.6225242764289213785e-003	-1.5277473123858904045e-003	-6.9183197562182804864e-004
-3.0447680255169362534e+001	-5.3177934960261367037e-001	9.0596584886274922101e+000
2.8177758090360373050e-004	-3.1469590804946202045e-003	-1.0794238049289112837e-003

Table 3: The initial conditions for the Nine Planets problem.

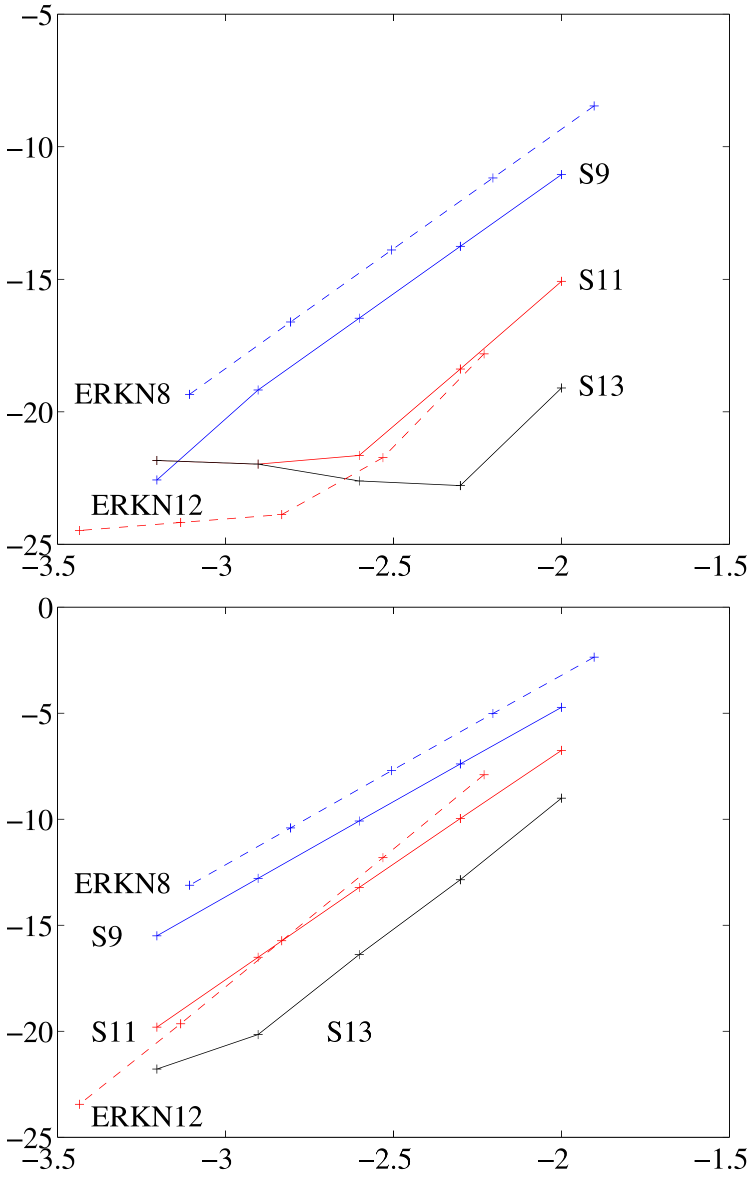

Figure 2: A log-log graph (base 10) of the maximum error in y_1 at $t = 10000/(2\pi)$ periods against the scaled stepsize for Kepler's problem, quadruple precision. Top: $e = 0$, bottom: $e = 0.5$.

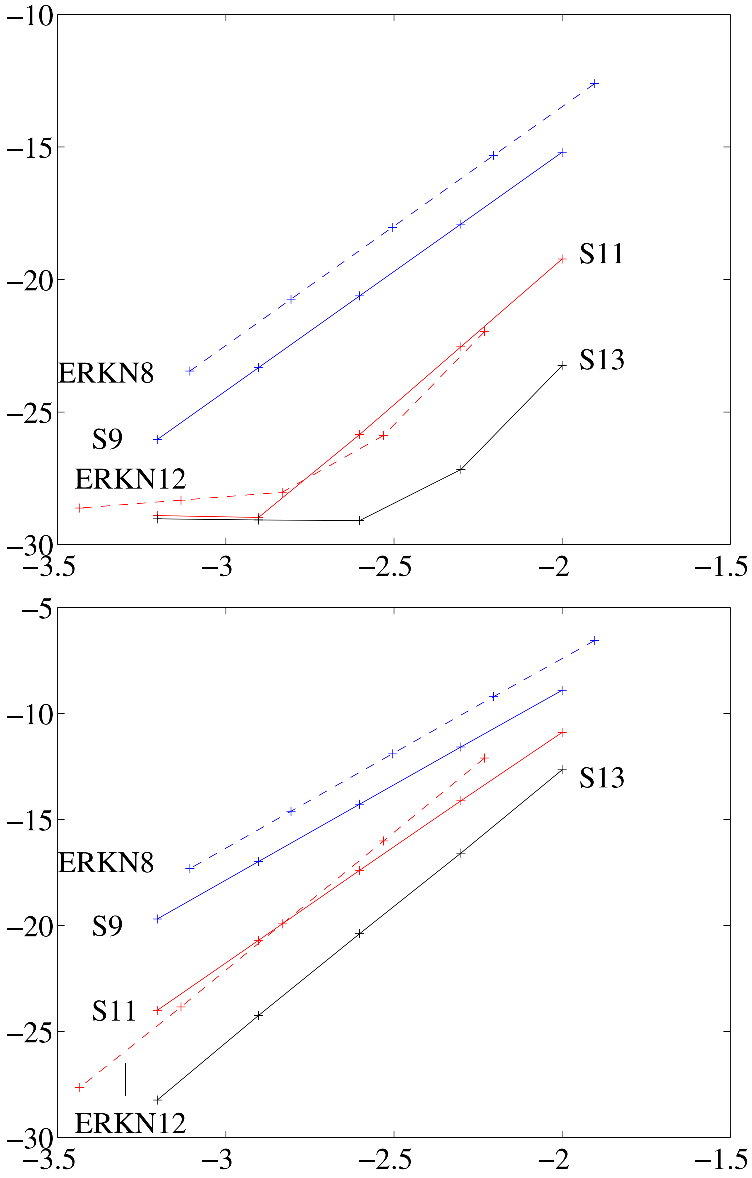

Figure 3: A log-log graph (base 10) of the maximum error in the relative energy against the scaled stepsize at $t = 10000/(2\pi)$ periods for Kepler's problem, quadruple precision. Top: $e = 0$, bottom: $e = 0.5$.

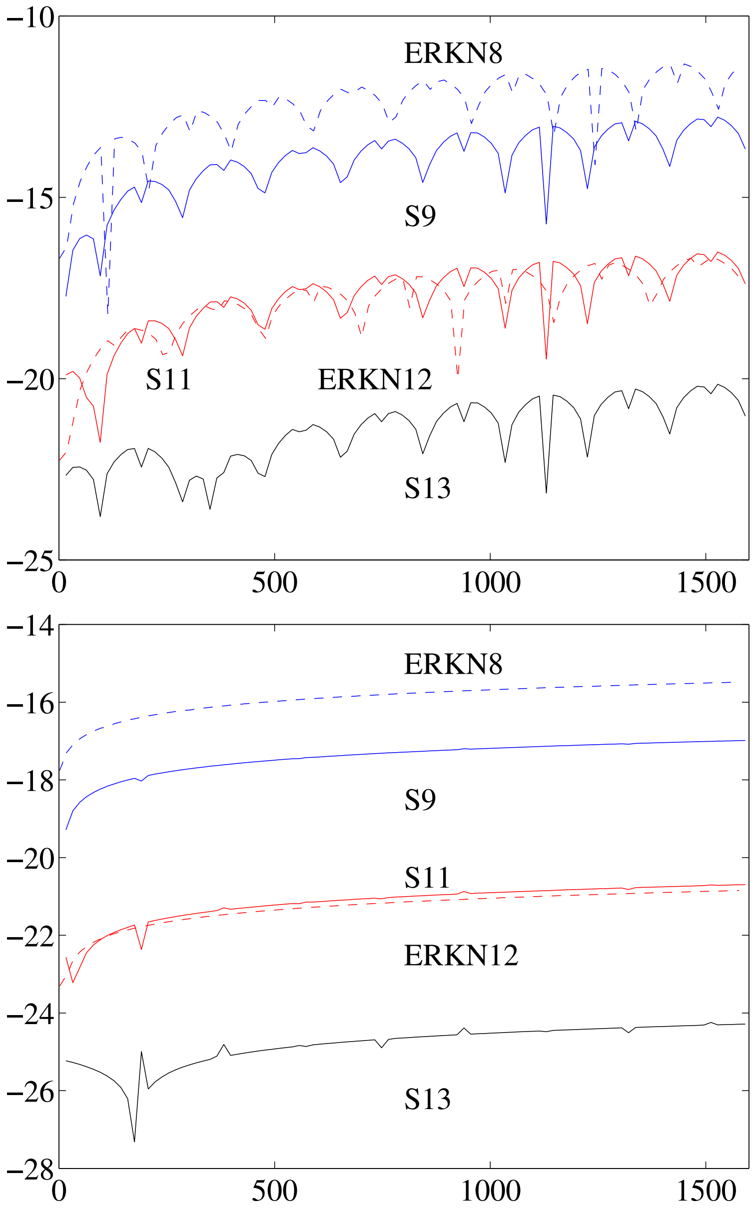

Figure 4: A semi-log graph (base 10) of the absolute error in y_1 (top) and the relative error in the energy (bottom) as a function of t for Kepler's problem with $e = 0.5$, quadruple precision. The scaled stepsize is 0.00125 and the time is in periods.

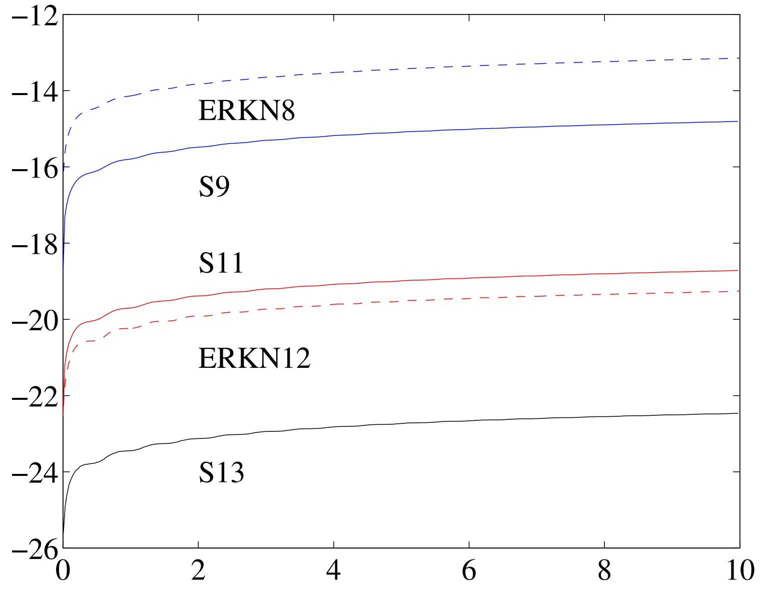

Figure 5: Semi-log graph (base 10) of the relative error in the total energy against t in units of 10^5 years for the Jovian problem, quadruple precision.

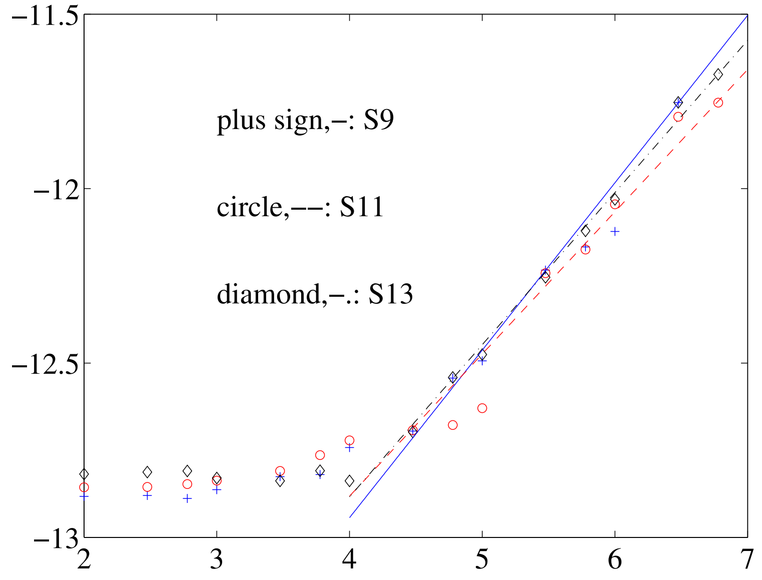

Figure 6: Log-log graph (base 10) of the relative error in the energy against t in years for S9, S11 and S13 applied to the Jovian problem, double precision.

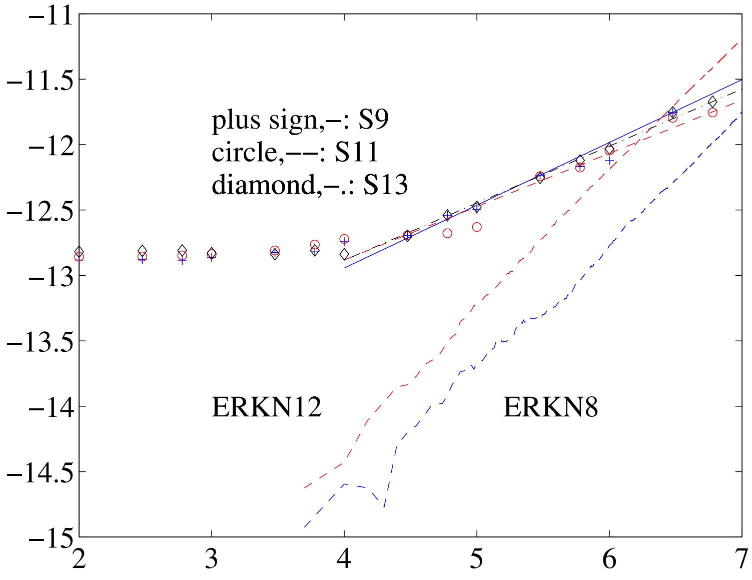

Figure 7: Log-log graph (base 10) of the relative error in the energy against t in years for S9, S11, S13, ERKN8 and ERKN12 applied to the Jovian problem, double precision.

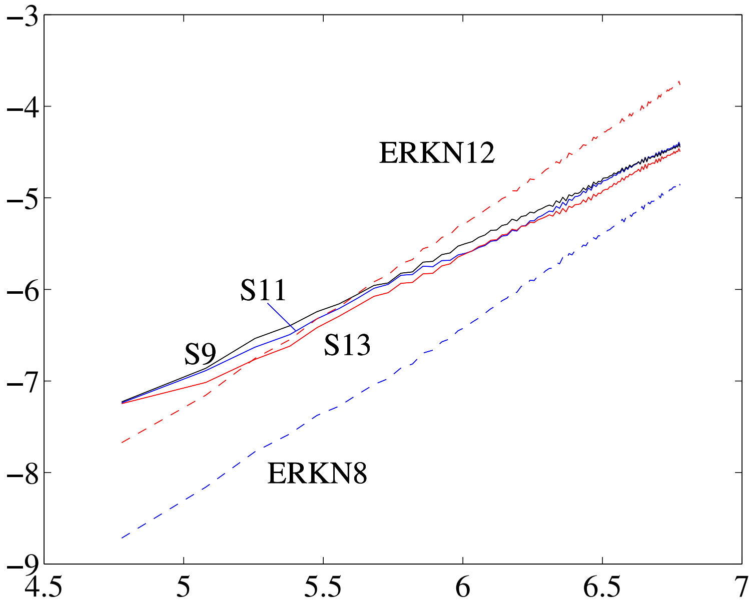

Figure 8: Log-log graph (base 10) of the error in the position of Jupiter against t in years for S9, S11, S13, ERKN8 and ERKN12 applied to the Jovian problem, double precision.

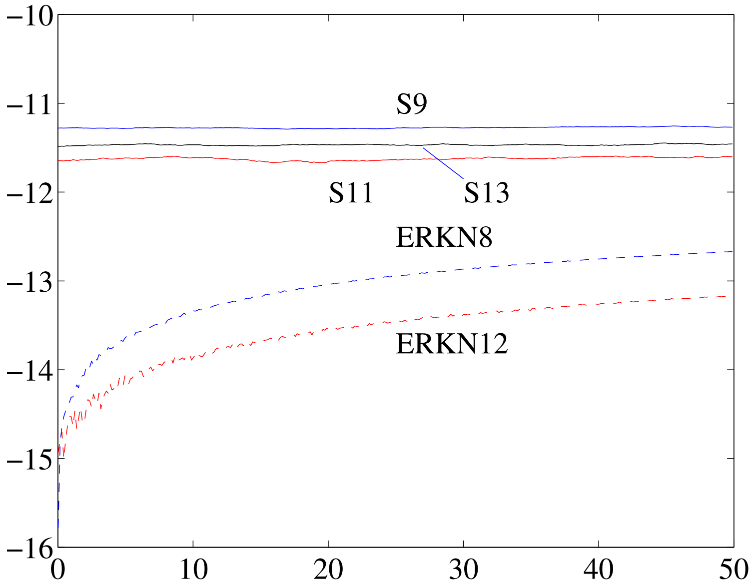

Figure 9: Log-log graph (base 10) of the relative error in the energy against t in units of one thousand years for S9, S11, S13, ERKN8 and ERKN12 applied to the Nine Planets problem, double precision.