PUBLISHED BY IOP PUBLISHING FOR SISSA

RECEIVED: *February 4, 2009*
REVISED: *March 6, 2009*
ACCEPTED: *March 9, 2009*
PUBLISHED: *April 1, 2009*

Track parameter propagation through the application of a new adaptive Runge-Kutta-Nyström method in the ATLAS experiment

E. Lund,[a,1] **L. Bugge,**[a] **I. Gavrilenko**[b,c] **and A. Strandlie**[a,d]

[a] *University of Oslo,*
PO Box 1072 Blindern, 0316 Oslo, Norway
[b] *European Organization for Nuclear Research,*
CERN CH-1211, Genève 23, Switzerland
[c] *P. N. Lebedev Institute of Physics,*
Leninskiy prospekt 53, Moscow, Russia
[d] *Gjøvik University College,*
PO Box 191 Teknologivn. 22, 2802 Gjøvik, Norway

E-mail: esben.lund@fys.uio.no

ABSTRACT: In this paper we study several fixed step and adaptive Runge-Kutta methods suitable for transporting track parameters through an inhomogeneous magnetic field. Moreover, we present a new adaptive Runge-Kutta-Nyström method which estimates the local error of the extrapolation without introducing extra stages to the original Runge-Kutta-Nyström method. Furthermore, these methods are compared for propagation accuracy and computing cost efficiency in the *simultaneous track and error propagation* (STEP) algorithm of the common ATLAS tracking software. The tests show the new adaptive Runge-Kutta-Nyström method to be the most computing cost efficient.

KEYWORDS: Simulation methods and programs, Pattern recognition, cluster finding, calibration and fitting methods, Data processing methods

[1] Corresponding author.

doi:10.1088/1748-0221/4/04/P04001

2009 JINST 4 P04001

Contents

2009 JINST 4 P04001

1 Introduction

Experimental particle physics is on the verge of a new era, heralded by the Large Hadron Collider being commissioned at the European Organization for Nuclear Research, CERN, located just outside Geneva, Switzerland. The LHC accelerator will collide protons at a center of mass energy of 14 TeV, opening up a new window for particle discoveries and precision measurements of existing theories. Particle detectors are located at four beam crossings along the LHC, one of which houses the ATLAS detector [1]. This is the largest of the LHC experiments, employing a great variety of detector and magnetic field technologies to identify a wide range of particles. The complex magnetic field and high collision rate, however, make the reconstruction of particle tracks very challenging. Things are further complicated by the relatively big amount of material within ATLAS, generating considerable disturbances to the particle tracks through material interactions such as energy loss and multiple scattering.

Particle trajectories are often parametrized with respect to surfaces crossed by the track. At ATLAS, there are five standard track parameters [2]; two related to the track position, another two giving the track direction, while the last is related to the particle charge and momentum. Track parameter propagation — which is the process of transporting track parameters and their associated covariance matrices through the magnetic field and material of the detector — is an important part of any track reconstruction algorithm, hence high accuracy and speed are essential to the propagation.

Historically, track parameter propagation has often been handled by the fixed step Runge-Kutta-Nyström method [3], which is designed to solve second-order differential equations, such

as the equation of motion of a particle in a magnetic field. The basic idea of this method is to divide the integration interval into steps and solve these independently in an iterative procedure. The solution to each step is estimated by evaluating the equation of motion at different points — often referred to as *stages* — along the step. Fixed step methods are, however, not optimal in case of a very inhomogeneous magnetic field, such as in the ATLAS detector. To accomodate the most challenging parts of the ATLAS magnetic field, the fixed step size has to be small, thereby wasting time in the smooth parts of the field. Adaptive methods, such as the Runge-Kutta-Fehlberg method [4], solve this problem. These adaptive methods, however, require at least one extra stage in every step compared to the fixed step methods, but the amount of steps taken during the propagation is often reduced significantly.

In this paper, we study several adaptive Runge-Kutta methods along with the classical fixed step Runge-Kutta (RK) and Runge-Kutta-Nyström (RKN) methods. We also introduce a new adaptive Runge-Kutta-Nyström method, which combines the simplicity of the fixed step method with the efficiency of the adaptive methods without introducing extra stages. This is the method employed by the STEP algorithm, which is used for both stand-alone and combined muon reconstruction — along with fast track simulation in the muon spectrometer — within the ATLAS tracking framework [5].

In section 2, we present the equation of motion along with some fixed step and adaptive Runge-Kutta methods for solving the track parameter propagation, followed by the new adaptive Runge-Kutta-Nyström method. In section 3, we give a short presentation of the STEP algorithm before validating and comparing the Runge-Kutta methods in section 4. Finally we present a brief conclusion in section 5.

Natural units ($\hbar = c = 1$) are used throughout this paper.

2 Integrating the equation of motion

There are many representations of the equation of motion of a charged particle moving through a magnetic field [6], primarily separated by the free spatial parameter, which is mostly dictated by the geometry of the experiment. For a fixed target experiment, involving tracks going mainly along the beam axis, using a fixed Cartesian coordinate system and z as the free parameter is often satisfactory. In a beam collision, particles move outwards from the interaction point in all directions, facilitating r as the free parameter, either in a fixed spherical or cylindrical coordinate system. However, the introduction of a magnetic field makes the integration along a fixed axis difficult and error prone. Using the arc length s and integrating along the path itself solves this problem.

Technically, the main difference between using z, r and s as the free parameter is the need to constantly check the distance to the destination surface when applying s as the free parameter. This is because the s propagation sets up a curvilinear coordinate system which follows the track at all times, whereas the z and r propagations use a static coordinate system defined by the detector geometry. The increased complexity and computing cost of using a curvilinear system is, however, outweighed by its robustness and ability to handle sharply bending tracks.

By using s as the free parameter, the equation of motion, given by the Lorentz force, becomes

$$\frac{d^2\boldsymbol{r}}{ds^2} = \frac{q}{p}\left(\frac{d\boldsymbol{r}}{ds} \times \boldsymbol{B}(\boldsymbol{r})\right) = \lambda(\boldsymbol{T} \times \boldsymbol{B}(\boldsymbol{r})) \tag{2.1}$$

2009 JINST 4 P04001

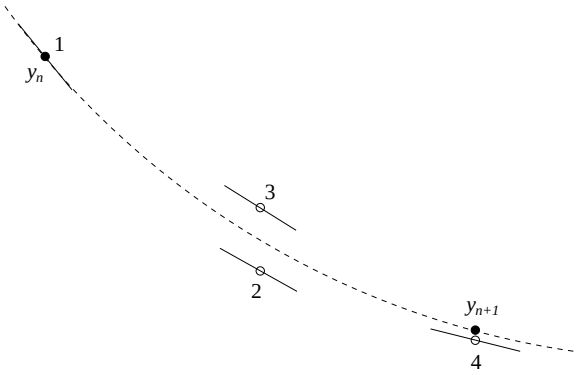

Figure 1. The classical fourth-order Runge-Kutta method. The derivative is evaluated four times in each step; once at the initial point, twice at trial midpoints and once at a trial endpoint. From these derivatives the final function value (shown as a filled dot) is calculated. (From ref. [7]).

where $\boldsymbol{T} = d\boldsymbol{r}/ds$ is the normalized tangent vector to the track, $\boldsymbol{B}(\boldsymbol{r})$ is the magnetic field and $\lambda \equiv q/p$. To see that $\boldsymbol{T}$ is the normalized tangent vector it is useful to look at the definition of the speed $d\boldsymbol{r}/dt = \boldsymbol{v}$, and the arc length $ds = vdt$. The derivative $d\boldsymbol{r}/ds$ then becomes $d\boldsymbol{r}/vdt = \boldsymbol{v}/v = \boldsymbol{T}$.

The electric field component of the Lorentz force is left out of the equation of motion because the electric field inside most of the detector is small compared to the magnetic field.

Given the inhomogeneous nature of the magnetic field, finding an analytical solution to the second-order differential equation of motion (2.1) becomes impossible. It has to be solved numerically. Many methods are available, but one group of methods proves especially useful in tracking. These are the recursive Runge-Kutta methods [7], most of which are developed for solving first-order differential equations, since higher order equations can always be reduced to sets of first-order equations by introducing auxiliary variables.

The basic idea of the recursive Runge-Kutta method is to divide the integration interval into steps and solve these independently in an iterative procedure. Every step becomes an initial value problem and can be solved as best suited for that particular part of the integration interval. This is especially useful when varying the step length to make the procedure adaptive. The solution to each step is estimated by evaluating the equation of motion at different points k — often referred to as *stages* — along the step. Every stage, except the first, is based on the previous stages of the step. In the end, all stages are weighted and summed to find the solution to the step. Figure 1 illustrates the four stages and the solution of the original Runge-Kutta method.

Generally speaking, every evaluation of the equation cancels an error term, producing a solution correct to the order of the number of evaluations. Hence, the four stages of the original Runge-Kutta method produce a result correct to the fourth order. Using higher-order solutions gives better accuracy per step and allows longer steps. However, if the integration environment is not known to the same order as the Runge-Kutta method, the extra evaluations may be better spent on more steps. Six evaluations of the equation of motion in a given interval can produce one sixth-order step, two third-order steps or three second-order steps. Which gives the best overall accuracy is hard to tell without testing. Later in this paper, it is shown that the ATLAS magnetic field does not warrant the use of Runge-Kutta methods beyond fourth order.

2009 JINST 4 P04001

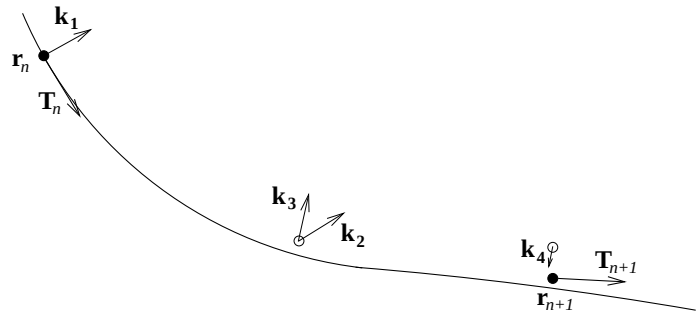

Figure 2. Integrating the equation of motion using the fourth-order Runge-Kutta-Nyström method. The equation of motion is evaluated four times in each step; once at the initial point, twice at a trial midpoint and once at a trial endpoint. From these stages $\boldsymbol{k}$, the position $\boldsymbol{r}_{n+1}$ and the normalized tangent vector to the track $\boldsymbol{T}_{n+1}$ are calculated.

2.1 The Runge-Kutta-Nyström method

These are the four stages and the solutions to the fourth-order Runge-Kutta-Nyström method;

$$\begin{aligned}
k_1 &= f(x_n, y_n, y'_n) \\
k_2 &= f\left(x_n + \frac{h}{2}, y_n + \frac{h}{2}y'_n + \frac{h^2}{8}k_1, y'_n + \frac{h}{2}k_1\right) \\
k_3 &= f\left(x_n + \frac{h}{2}, y_n + \frac{h}{2}y'_n + \frac{h^2}{8}k_1, y'_n + \frac{h}{2}k_2\right) \\
k_4 &= f\left(x_n + h, y_n + hy'_n + \frac{h^2}{2}k_3, y'_n + hk_3\right) \\
y'_{n+1} &= y'_n + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4) \\
y_{n+1} &= y_n + hy'_n + \frac{h^2}{6}(k_1 + k_2 + k_3)
\end{aligned} \tag{2.2}$$

where h is the step length, x is the integration variable, n is the step number and f is the second-order differential equation

$$\frac{d^2y(x)}{dx^2} = f(x, y, y') \tag{2.3}$$

which is integrated twice; first to produce the y'_{n+1} solution, second to produce y_{n+1}. This method was specifically designed for solving second-order differential equations such as the equation of motion. Integrating the equation of motion (2.1), figure 2, the first integration gives the normalized tangent vector $\boldsymbol{T}_{n+1}$, while the second integration gives the position of the particle $\boldsymbol{r}_{n+1}$.

An appealing aspect of the Runge-Kutta-Nyström method is that stage two and three share the same point in space; $(x_n + \frac{h}{2}, y_n + \frac{h}{2}y'_n + \frac{h^2}{8}k_1)$, and hence magnetic field. To reduce calls to the magnetic field service further, it is possible to use the magnetic field found in the last stage of the current step as the starting field of the next step, because the position of the last stage $(x_n + h, y_n + hy'_n + \frac{h^2}{2}k_3)$ is very close to the final solution of the step $(x_n + h, y_n + hy'_n + \frac{h^2}{6}(k_1 + k_2 + k_3))$, i.e. the starting point of the next step. By using these optimizations, the Runge-Kutta-Nyström method only needs to evaluate the magnetic field twice per step.

2009 JINST 4 P04001

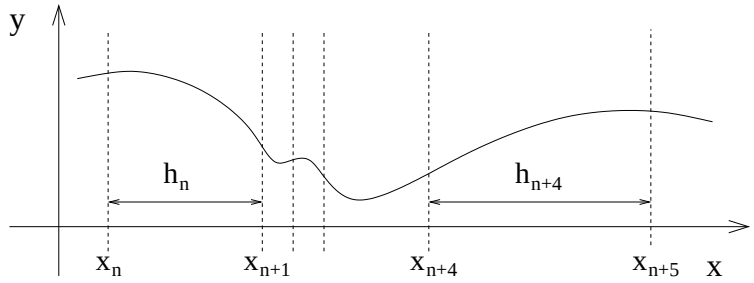

Figure 3. Illustration of the varying step size of the adaptive Runge-Kutta methods. The step size algorithm adjusts to the roughness of the function by using small steps in the rocky sections and longer steps in the smooth parts.

It is, however, a problem of the Runge-Kutta-Nyström method that it only gives an idea of the quality of the integration, i.e. correct to the fourth order. The actual integration error of the step is unknown. Furthermore, the accuracy of the integration can only be adjusted by setting the fixed step length h, which has to accommodate the most inhomogeneous parts of the magnetic field. In ATLAS, this step length is a lot smaller than that needed for the more well-behaved parts of the field, making a fixed step length inefficient.

2.2 Adaptive Runge-Kutta methods

To avoid the problems of the fixed step length methods, and to adjust the accuracy in a predictable way, an adaptive Runge-Kutta method is needed. The most straightforward technique is step doubling. Every step is taken twice, first as a full step, and then as two half-steps producing a solution of higher order. The difference between these solutions gives an estimate of the *local error* ϵ of each step of the lower-order full step solution. The local error is checked against a user specified criterion, the *error tolerance* τ, and if the error is bigger than the error tolerance the step fails. It is then shortened and redone. If the local error is smaller than the error tolerance, the step is successful and the step size of the following step is increased to minimize the number of steps needed for the integration. A good step size selection algorithm provides *error tolerance proportionality*, which means that the logarithm of the global error of the whole integration is proportional to the logarithm of the user specified error tolerance. This is achieved by keeping the local error close to the error tolerance during the integration by adjusting the step size, figure 3. Managing this with a minimum number of failed steps is the main challenge of the step size selection algorithm.

2.3 Embedded Runge-Kutta pairs

An alternative way of estimating the local error of a step is based on the embedded Runge-Kutta formulae — the so-called embedded pairs — invented by Fehlberg [4]. The most well-known Runge-Kutta-Fehlberg method is a fifth-order method with six stages, where another combination of these six stages gives a fourth-order method. As in step doubling, the difference between these two solutions is used to estimate the local error of the lower-order solution of the step. Since the embedded formulae provide error estimation at almost no extra cost, many additional embedded pairs of varying orders have been created since Fehlberg's original work, most of which contain only

2009 JINST 4 P04001

one extra stage compared to the classical Runge-Kutta methods of the same order. The Bogacki-Shampine 3(2)FSAL (BS32) embedded pair [8] provides an illustrative example;

$$\begin{aligned}
k_1 &= hf(x_n, y_n) \\
k_2 &= hf\left(x_n + \frac{h}{2}, y_n + \frac{k_1}{2}\right) \\
k_3 &= hf\left(x_n + \frac{3}{4}h, y_n + \frac{3}{4}k_2\right) \\
k_4 &= hf\left(x_n + h, y_n + \frac{2}{9}k_1 + \frac{1}{3}k_2 + \frac{4}{9}k_3\right) \\
\text{3rd-order solution: } y_{n+1} &= y_n + \frac{2}{9}k_1 + \frac{1}{3}k_2 + \frac{4}{9}k_3 \\
\text{2nd-order solution: } \hat{y}_{n+1} &= y_n + \frac{7}{24}k_1 + \frac{1}{4}k_2 + \frac{1}{3}k_3 + \frac{1}{8}k_4
\end{aligned} \tag{2.4}$$

This is a "first-same-as-last" (FSAL) pair, meaning that the last stage k_4 is evaluated at the final higher-order solution of the step y_{n+1}. Since the next step starts out where the current ends, we get $k_1(n+1) = k_4(n)$, hence the name "first-same-as-last". This property reduces the number of evaluations of the differential equation, thereby increasing the efficiency. The 3(2) in the naming of the pair indicates the orders of the pair, the lower-order solution $\hat{y}_{n+1}$ being used for error estimation only.

Since the error estimate applies to the lower-order solution, the higher-order solution is often more accurate than the error estimate suggests. It has therefore become a standard procedure to use the higher-order solution along with the error estimate of the lower-order solution unless very precise error estimates are needed. This is the so-called local extrapolation, which is applied throughout this paper.

2.4 Adjusting the step size according to the error tolerance

As mentioned earlier, a step size selection algorithm using the local error estimate of each step

$$\epsilon = y_{n+1} - \hat{y}_{n+1} \tag{2.5}$$

is needed to make the Runge-Kutta methods adaptive. The most common step size algorithm is

$$h_{n+1} = h_n \left(\frac{\tau}{|\epsilon|}\right)^{\frac{1}{q+1}} \tag{2.6}$$

where the new step length h_{n+1} is given by the local error of the current step ϵ, the current step length h_n, the user specified error tolerance τ, and the order q of the lower-order solution $\hat{y}_{n+1}$. The derivation of this algorithm is left to refs. [7, 9]. The core of the algorithm is the fraction $\tau/|\epsilon|$ which becomes less than one if the local error is bigger than the error tolerance, reducing the step length, and vice versa.

After "trimming" by a limitation criterion

$$\frac{1}{4}h_n \leq h_{n+1} \leq 4h_n \tag{2.7}$$

2009 JINST 4 P04001

the new step size h_{n+1} is used for the next step, or for retrying the current step if it fails. The limitation criterion is introduced to prevent extreme changes to the step size during the integration. Without the limitation criterion, a tiny bump or smooth region in the integration interval may change the step length by a factor of hundred or more. This may compromise the stability of the solution and the error tolerance proportionality.

After finding the new step length (2.6) and trimming it (2.7), the quality of the current step is evaluated by an acceptance criterion

$$|\epsilon| < 4\tau \tag{2.8}$$

to decide whether the step is accepted. The safety factor of 4 is introduced to allow for the local error to oscillate around the optimal value of $|\epsilon| = \tau$.

Given the above step size algorithm (2.6) and acceptance criterion (2.8), the estimated global error of the integration becomes

$$g_\epsilon = c\tau^a \tag{2.9}$$

For a given Runge-Kutta method and differential equation, c and a are constants. The logarithm of the global error g_ϵ — and hence integration accuracy — is therefore proportional to the logarithm of the user specified error tolerance. The error tolerance proportionality of the global error is very useful in the reconstruction software, allowing the same propagator to be used for a slow, accurate reconstruction, or a fast, less precise high-level trigger by only adjusting the error tolerance.

2.5 An adaptive Runge-Kutta-Nyström method

The fourth-order Runge-Kutta-Nyström method (2.2) is in many ways the simplest and most attractive of the Runge-Kutta methods discussed above. Estimating the local error of each Runge-Kutta-Nyström step — thereby enabling adaptivity through step size adjustment — is basically all that separates this fixed step method from the adaptive methods. In this section, we present a novel approach to estimating this local error, thus enabling the creation of an adaptive Runge-Kutta-Nyström method through the application of the formulae described in section 2.4.

To estimate the local error of a Runge-Kutta-Nyström step it is useful to look at the Taylor expansion around x_n that forms the basis of the Runge-Kutta methods;

$$g(x_n + h) = g(x_n) + hg'(x_n) + \frac{1}{2}h^2 g''(x_n) + \frac{1}{6}h^3 g^{(3)}(x_n) + \frac{1}{24}h^4 g^{(4)}(x_n) + \cdots \tag{2.10}$$

The main part of the local error is the truncation error from higher-order terms that are excluded, i.e. all terms above fourth order for the Runge-Kutta-Nyström method. This leaves out the possibility to calculate the local error directly, but it makes sense to assume that the last known term indicates how quickly the Taylor expansion converges. A small term indicates a quick convergence and small truncation (local) error, whereas a big term indicates the opposite. This assumption is confirmed by the widely used local error estimate (2.5) of the embedded pairs. For a 4(3) pair, the local error becomes

$$\begin{aligned} &\epsilon = y_{n+1} - \hat{y}_{n+1} = g(x_n + h)_{\text{fourth order}} - g(x_n + h)_{\text{third order}} = \\ &g(x_n + h)_{\text{fourth-order term}} = \tfrac{1}{24}h^4 g^{(4)}(x_n) \end{aligned} \tag{2.11}$$

2009 JINST 4 P04001

which is the last known term. Calculating this term directly from the four stages of the Runge-Kutta-Nyström method is not possible. However, since any error estimate along the step will suffice, we can calculate it in the middle of the step ($x_n + \frac{1}{2}h$), instead of at the beginning (x_n). This allows us to find the local error ϵ by using the symmetric derivative

$$g'(x_n) \approx \frac{g(x_n + h) - g(x_n - h)}{2h} \tag{2.12}$$

and the four stages given by the second-order derivatives k_l ($= g_l''$) already calculated by the Runge-Kutta-Nyström method.

First, we find the third-order symmetric derivative of the first half of the step by using a step size of $\frac{1}{4}h$, figure 4,

$$g^{(3)}(x_n + \frac{1}{4}h) = \frac{g''(x_n + \frac{1}{4}h + \frac{1}{4}h) - g''(x_n + \frac{1}{4}h - \frac{1}{4}h)}{\frac{1}{2}h} = \frac{g_2''(x_n + \frac{1}{2}h) - g_1''(x_n)}{\frac{1}{2}h} = \frac{k_2 - k_1}{\frac{1}{2}h} \tag{2.13}$$

Moreover, we calculate the third-order symmetric derivative of the last half of the step. Again, using a step size of $\frac{1}{4}h$,

$$g^{(3)}(x_n + \frac{3}{4}h) = \frac{g''(x_n + \frac{3}{4}h + \frac{1}{4}h) - g''(x_n + \frac{3}{4}h - \frac{1}{4}h)}{\frac{1}{2}h} = \frac{g_4''(x_n + h) - g_3''(x_n + \frac{1}{2}h)}{\frac{1}{2}h} = \frac{k_4 - k_3}{\frac{1}{2}h} \tag{2.14}$$

These two derivatives are then used to calculate the fourth-order symmetric derivative at the middle of the step

$$g^{(4)}(x_n + \frac{1}{2}h) = \frac{g^{(3)}(x_n + \frac{1}{2}h + \frac{1}{4}h) - g^{(3)}(x_n + \frac{1}{2}h - \frac{1}{4}h)}{\frac{1}{2}h} = \frac{g^{(3)}(x_n + \frac{3}{4}h) - g^{(3)}(x_n + \frac{1}{4}h)}{\frac{1}{2}h} = \frac{\frac{k_4 - k_3}{\frac{1}{2}h} - \frac{k_2 - k_1}{\frac{1}{2}h}}{\frac{1}{2}h} = \frac{k_1 - k_2 - k_3 + k_4}{\frac{1}{4}h^2} \tag{2.15}$$

corresponding to the error estimate

$$\epsilon = \frac{1}{24}h^4 g^{(4)}(x_n + \frac{1}{2}h) = \frac{1}{24}h^4 \frac{k_1 - k_2 - k_3 + k_4}{\frac{1}{4}h^2} = \frac{h^2}{6}(k_1 - k_2 - k_3 + k_4) \tag{2.16}$$

Since we are always looking at the local error estimate relative to some arbitrarily chosen error tolerance (2.6), (2.8), we can absorb the $1/6$ into the error tolerance, leaving only

$$\epsilon = h^2(k_1 - k_2 - k_3 + k_4) \tag{2.17}$$

Whereas the embedded Runge-Kutta pairs introduce extra stages to estimate the local error, this approach only recycles the four Runge-Kutta-Nyström stages already known.

2009 JINST 4 P04001

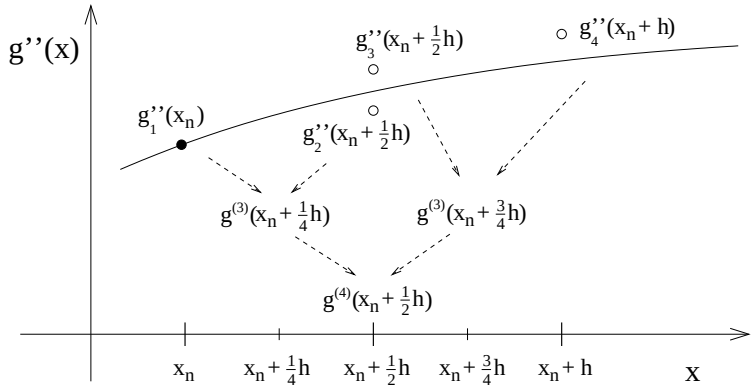

Figure 4. Using the four stages k_l $(= g_l'')$ of the fourth-order Runge-Kutta-Nyström method to estimate the local error ϵ of a step. The four stages are used to produce the third-order symmetric derivatives at the intermediate points $(x_n + \frac{1}{4}h)$ and $(x_n + \frac{3}{4}h)$, which go into the calculation of the fourth-order symmetric derivative in $(x_n + \frac{1}{2}h)$, corresponding to the local error ϵ.

Equipped with the local error estimate (2.17), we can apply the step size algorithm (2.6) with $q = 3$ — assuming an error estimate correct to one order less than the Runge-Kutta-Nyström method — followed by the limitation criterion (2.7) and the acceptance criterion (2.8) to create an adaptive Runge-Kutta-Nyström method. Thus, adaptivity is achieved by following the approach of the embedded Runge-Kutta pairs, only the local error estimate is new. In the following sections, we show that this adaptive Runge-Kutta-Nyström method is very computing cost efficient, and hence has become the standard STEP method.

3 The STEP algorithm

The STEP algorithm transports the track parameters from one surface of the tracking geometry to another. To find the crossing point with the target surface, the algorithm starts off with finding the distance from the starting point to the surface in the direction of the momentum. If it is less than 10 cm the propagation is done in a single step, else the propagation starts off with a 10 cm step. The STEP algorithm then does the initial step applying the chosen Runge-Kutta method. From the step size and local error estimate of the first step, a new step size is calculated by the step size algorithm. This new step size is then "trimmed" according to the limitation criterion before a decision to go on, or retry the current step is made by the acceptance criterion.

The whole procedure, starting with finding the distance to the target surface is then repeated. This time, and for all the following steps, the distance is only used as a maximum limit to the new step size to avoid stepping through the target surface. When the distance to the target surface is down to 10 μm, or less, the Runge-Kutta propagation is stopped, leaving the remaining propagation to a simple Taylor expansion

$$\begin{aligned} \boldsymbol{r}_{n+1} &= \boldsymbol{r}_{\mathrm{RK}} + h\boldsymbol{r}'_{\mathrm{RK}} + \tfrac{1}{2}h^2\boldsymbol{r}''_{\mathrm{RK}} = \boldsymbol{r}_{\mathrm{RK}} + h(\tfrac{d\boldsymbol{r}}{ds})_{\mathrm{RK}} + \tfrac{1}{2}h^2(\tfrac{d^2\boldsymbol{r}}{ds^2})_{\mathrm{RK}} \\ \boldsymbol{T}_{n+1} &= \boldsymbol{T}_{\mathrm{RK}} + h\boldsymbol{T}'_{\mathrm{RK}} \qquad\qquad = (\tfrac{d\boldsymbol{r}}{ds})_{\mathrm{RK}} + h(\tfrac{d^2\boldsymbol{r}}{ds^2})_{\mathrm{RK}} \end{aligned} \tag{3.1}$$

2009 JINST 4 P04001

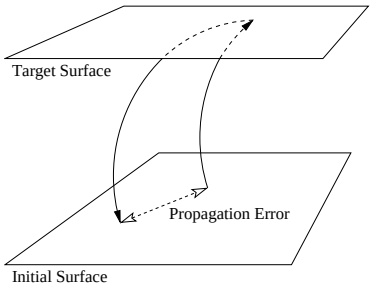

Figure 5. The definition of the propagation error in the test setup. Dividing the error by the total path length — back and forth — produces the relative propagation error.

where h is the remaining distance to the target surface and the derivatives are given by the equation of motion (2.1). The parameters indexed by RK are those found by the last step of the Runge-Kutta propagation.

Since the standard STEP algorithm uses an adaptive — instead of a fixed step — Runge-Kutta method, the initial step length is of less importance, and hence cannot be set by the user.

4 Validating the parameter propagation

To test the STEP algorithm and compare Runge-Kutta methods, we propagate the same set of charged particles through the realistic ATLAS magnetic field by using the different methods. The particles are sent in random directions, covering all azimuthal and polar angles at momenta ranging from 500 MeV to 500 GeV, starting off from an initial surface located at the center of the ATLAS detector, figure 5. The target surface is randomly placed and rotated in a cube with sides of 20 m centered in the detector. Upon reaching the target surface, the particles are propagated straight back to the initial surface, which they should ideally hit exactly where they originally started out. However, due to errors introduced by the propagation, the final position is slightly displaced, giving the global propagation error of the track. Due to the great variation of the distance to the target surface, the global propagation error is divided by the total path length — back and forth — to produce a global propagation error per unit distance. To avoid smearing of the global propagation errors by the orientation of the initial surface, it is always set normal to the initial particle momentum. For simplicity, the global propagation error per unit distance is hereafter referred to as the *relative propagation error*.

This test does not represent typical ATLAS situations in the sense that the actual propagation distances are not randomly distributed in the detector volume; the majority being short range propagations in the inner detector. Furthermore, the particle momenta are taken at random from a flat distribution — as opposed to the fixed momenta often used in such tests — to produce a quick and easy way of covering a wide range of momenta. This approach is acceptable since the propagation errors depend heavily on external factors specific to the experiment, such as the magnetic field, limiting the relevance of studying propagation errors at specific momenta. In spite of these short cuts — and because the same random set of particles are used by all methods — this test is sufficient for comparing and qualifying propagation algorithms.

2009 JINST 4 P04001

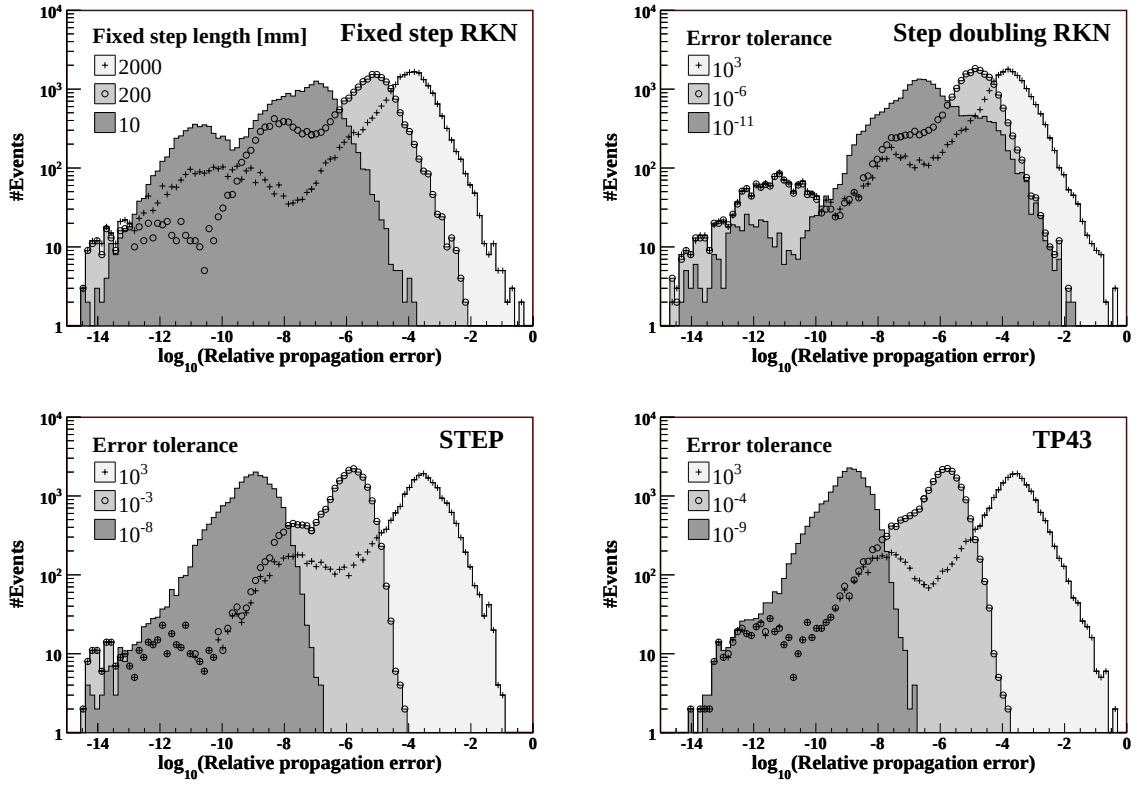

Figure 6. Distributions of the logarithms of the relative propagation errors of the fixed step, step doubling and adaptive (STEP) Runge-Kutta-Nyström methods, and for the embedded pair TP43, for three fixed step lengths and error tolerances.

4.1 Relative propagation errors

Figure 6 shows distributions of the logarithms of the relative propagation errors at different fixed step lengths and error tolerances. Distributions are presented for the fixed step, step doubling and adaptive (STEP) Runge-Kutta-Nyström methods, and for the embedded pair Tsitouras-Papakostas 4(3)FSAL (TP43) [9]. In this pair, the last two stages are both calculated at the full step length, making them closely positioned in space, almost sharing magnetic field values. Here, we use identical field values for both stages to increase the efficiency of the pair. The classical Runge-Kutta distributions (not shown) are similar to those of the Runge-Kutta-Nyström method, both in the fixed step and step doubling case.

The distributions of figure 6 show several peaks; one at -12 which is made up of very short tracks, typically 10 cm. Another at -8 which is made up of tracks within the ATLAS solenoid, and the main peak at -4 which contains the tracks moving into the ATLAS toroidal magnetic field, which is the most challenging part to maneuver accurately. In most cases, a relative propagation error below 10^{-6} is sufficient.

Figure 7 shows the mean relative propagation errors as a function of the error tolerance for the above-mentioned adaptive Runge-Kutta methods, along with three other embedded pairs; Cash-Karp 5(4) (CK54) [7], Dormand-Prince 5(4)FSAL (DP54) [10] and Tsitouras-Papakostas 7(5)

2009 JINST 4 P04001

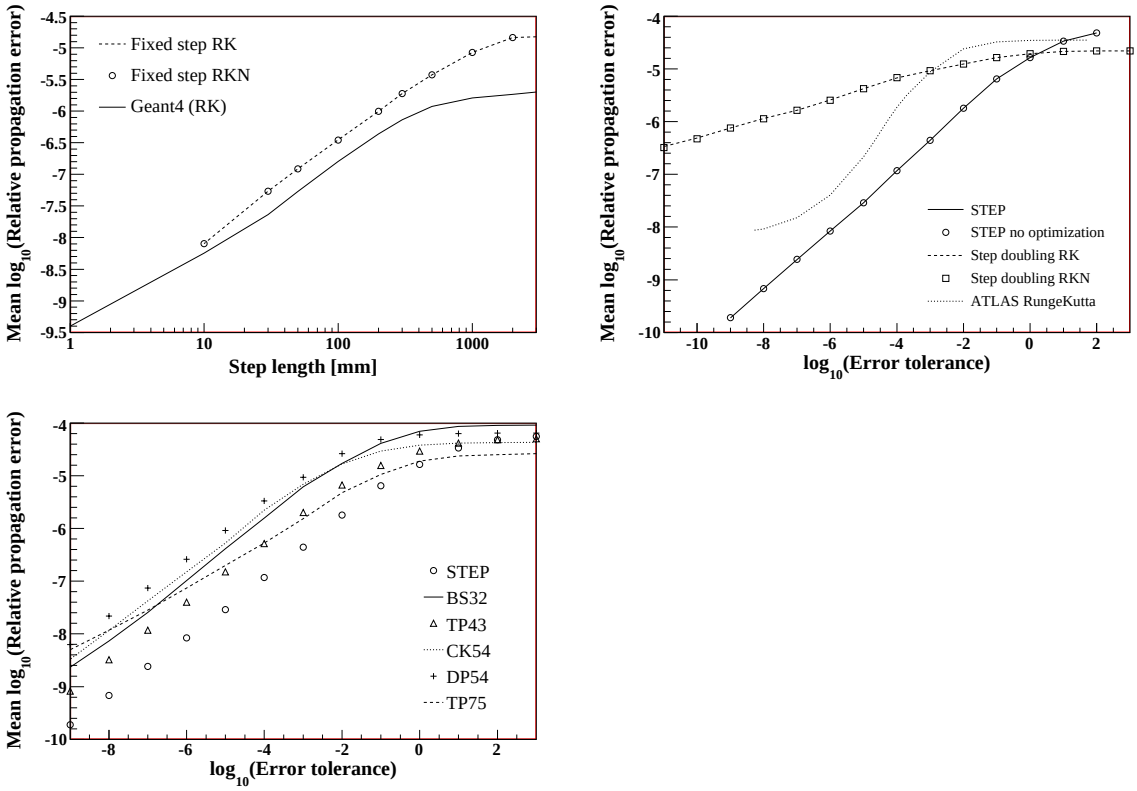

Figure 7. Mean values of the logarithms of the relative propagation errors of some fixed step and adaptive Runge-Kutta methods.

(TP75) [9]. Similarly to the TP43 pair, the last two stages of DP54 are both calculated at the full step length, sharing the same magnetic field values to increase the efficiency of the pair. Furthermore, the fixed step methods are presented as functions of the step length. The "ATLAS RungeKutta" is another ATLAS propagator — originally part of the xKalman package [11] — while GEANT4 [12] is a simulation package developed for tracking particles through material.

As mentioned in section 2.1, it is possible to use the same magnetic field values for calculating the last stage of the current step and the first stage of the next step in the Runge-Kutta-Nyström method because both stages are closely located in space. This optimization is implemented in the adaptive Runge-Kutta-Nyström method employed by STEP, and it is validated by the "STEP no optimization" curve of the top left plot of figure 7.

In figure 7, we see that all of the methods reach a plateau at high error tolerance, which is the inherent accuracy of the method. On the plateau, the error control is practically disabled and the propagation goes "unchecked", similarly to the classical fixed step methods. Besides, the computing time also levels out, gaining nothing by going to very high error tolerances.

The linearity of figure 7 at error tolerances below 10^{-1} — where the error control is in full effect — clearly shows the error tolerance proportionality of the relative propagation error (2.9). This allows for steering the integration accuracy in a predictable way through adjusting the error tolerance. The constants of eq. (2.9) can easily be found from figure 7, however, since these

2009 JINST 4 P04001

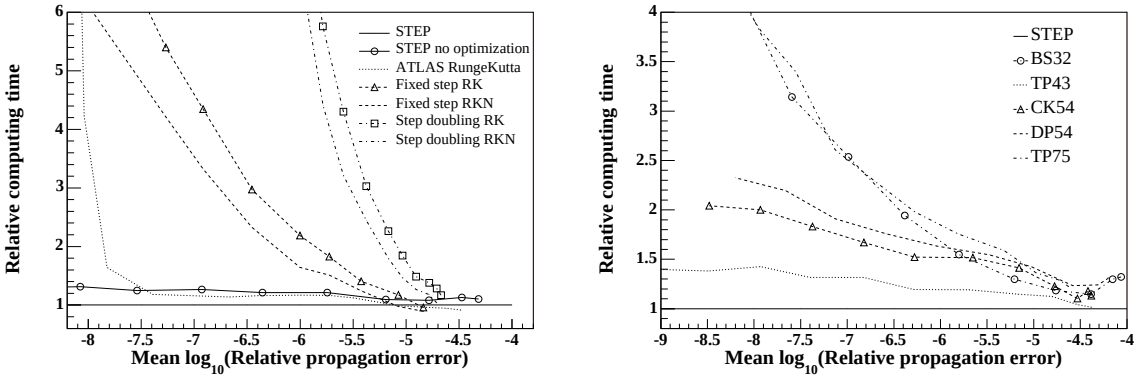

Figure 8. Computing cost efficiencies relative to the efficiency of the adaptive Runge-Kutta-Nyström method (STEP).

constants rely heavily on the experimental setup, they are of little interest beyond the ATLAS experiment, and hence are left out of this paper.

Surprisingly, the step doubling methods perform worse than the fixed step methods, showing little correlation between the error tolerance and the relative propagation error. This is because halving the step length does not necessarily increase the accuracy of the step sufficiently to give a good local error estimate. Moreover, in many cases the local error estimate of the step doubling is too low, giving an overly optimistic view of the quality of the propagation. This prevents the step size algorithm from working properly, making the steps too long to produce the required accuracy. These tracks show up as an extra shoulder on the right-hand side of the step doubling distribution of the lowest error tolerance of figure 6. Trials by using step tripling give a slightly better error estimate, but still worse than that of the other adaptive methods.

4.2 Efficiencies

All of the above-mentioned Runge-Kutta methods can produce a sufficient relative propagation error, even though the error tolerances needed to produce the desired accuracy differs. To compare methods, we need to study the computing cost of reaching some given accuracy; the so-called efficiency. Figure 8 shows efficiencies relative to the most efficient method, the new adaptive Runge-Kutta-Nyström method (STEP). From the plot on the left, we see that the optimized STEP method is slightly more efficient than the non-optimized STEP method. Moreover, we see that the fixed step methods are outperformed by the adaptive methods, especially at high accuracy (small propagation errors). To achieve such accuracy, the fixed steps must be limited to a few cm each, slowing the propagation down significantly. Furthermore, we notice that the step doubling methods perform surprisingly bad because every step is actually made up of three steps; two steps for the solution, and one for the error estimation, producing a significant overhead for failed steps.

From the plot on the right of figure 8, we see that the embedded pairs — especially TP43 — fare better than the fixed step and step doubling methods. The efficiency of the embedded pairs depends heavily on how well they are matched to the underlying problem; defined by the equation of motion and the ATLAS magnetic field. The fourth-order STEP and TP43 methods are clearly best suited in this case. The fifth-order CK54 and DP54 pairs also fare reasonably well, whereas

2009 JINST 4 P04001

the lowest (BS32) and highest (TP75) order pairs fail at high accuracy. This probably happens because BS32 struggles to produce the required accuracy, whereas TP75 becomes more meticulous than necessary.

Even though the new adaptive Runge-Kutta-Nyström method is the most efficient in ATLAS, the TP43 efficiency becomes comparable when the computing cost of polling the magnetic field is negligible, such as in the case of a parameterized magnetic field.

5 Conclusion and outlook

In this paper, we have presented a new adaptive Runge-Kutta-Nyström method developed for the STEP algorithm within the ATLAS tracking framework. The STEP algorithm transports track parameters and their associated covariance matrices through the ATLAS detector, taking the magnetic field and material interactions into account. A relatively extensive computing cost comparison with existing numerical methods — suited for parameter propagation — shows the new adaptive Runge-Kutta-Nyström method to be the most efficient, having potential beyond the STEP application presented here.

In addition to the track parameter propagation discussed here, the transport of the associated covariance matrix — the so-called error propagation — is necessary to provide a full description of the track. The covariance matrix provides information about the uncertainties related to the reconstruction of the track parameters from the empirical data, and its transport is treated in a separate paper [13].

Acknowledgments

This work has been carried out as part of the developments of the ATLAS tracking group. We would like to thank our colleagues for their help in integrating the software and for their support in preparing this note.

References

[1] ATLAS collaboration, G. Aad et. al., *The ATLAS Experiment at the CERN Large Hadron Collider*, 2008 *JINST* **3** S08003.

[2] F. Åkesson et al., *The ATLAS Tracking Event Data Model*, ATLAS Public Note ATL-SOFT-PUB-2006-004 (2006).

[3] E.J. Nyström, *Über die numerische Integration von Differentialgleichungen*, *Acta Soc. Sci. Fenn.* **50** (1925) 1.

[4] E. Fehlberg, *Classical fifth-, sixth-, seventh-, and eighth-order Runge-Kutta formulas with stepsize control*, NASA-TR-R-287 (1968).

[5] T. Cornelissen et al., *Concepts, Design and Implemenentation of the ATLAS New Tracking (NEWT)*, ATLAS Public Note ATL-SOFT-PUB-2007-007 (2007).

[6] L. Bugge and J. Myrheim, *Tracking and track fitting*, *Nucl. Instrum. Meth.* **179** (1981) 365.

[7] W. Press et al., *Numerical Recipes in C*, Cambridge University Press, Cambridge U.K. (1999).

2009 JINST 4 P04001

[8] P. Bogacki and L.F. Shampine, *A 3(2) pair of Runge-Kutta formulae*, *Appl. Math. Lett.* **2** (1989) 321.

[9] C. Tsitouras and S.N. Papakostas, *Cheap error estimation for Runge-Kutta methods*, *SIAM J. Sci. Comput.* **20** (1999) 2067.

[10] J.R. Dormand and P.J. Prince, *A family of embedded Runge-Kutta formulae*, *J. Comp. Appl. Math.* **6** (1980) 19.

[11] I. Gavrilenko, *Description of Global Pattern Recognition Program (XKALMAN)*, ATLAS Note ATL-INDET-97-165 (1997).

[12] GEANT4 collaboration, S. Agostinelli et. al., *GEANT4: a simulation toolkit*, *Nucl. Instrum. Meth.* **A 506** (2003) 250.

[13] E. Lund et al., *Transport of covariance matrices in the inhomogeneous magnetic field of the ATLAS experiment by the application of a semi-analytical method*, ATLAS Public Note ATL-SOFT-PUB-2009-002, submitted to JINST (2009).

2009 JINST 4 P04001