PUBLISHED BY IOP PUBLISHING FOR SISSA

RECEIVED: *February 4, 2009*
ACCEPTED: *April 2, 2009*
PUBLISHED: *April 24, 2009*

Transport of covariance matrices in the inhomogeneous magnetic field of the ATLAS experiment by the application of a semi-analytical method

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 the transport of track parameter covariance matrices — the so-called error propagation — in the inhomogeneous magnetic field of the ATLAS experiment. The Jacobian elements are transported in parallel with the track parameters, avoiding the inherent need of any purely numerical scheme of propagating a set of auxiliary tracks. We evaluate the quality of the transported Jacobians by a very thorough, purely numerical approach of obtaining the same derivatives, providing a quantitative understanding of the effects of including gradients of energy loss and the magnetic field on the accuracy of the error propagation. Irrespective of the accuracy of the underlying track parameter propagation, the method of parallel integration of the derivatives is demonstrated to be significantly faster than even the simplest numerical scheme. The error propagation presented in this paper is part of the *simultaneous track and error propagation* (STEP) algorithm of the common ATLAS tracking software.

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

[1] Corresponding author.

doi:10.1088/1748-0221/4/04/P04016

Contents

1 Introduction

Experimental particle physics is opening a new window for particle discoveries and precision measurements of existing theories by the startup of 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 at four beam crossings, one of which houses the ATLAS detector [1]. This is the largest of the LHC experiments, employing a great variety of detector technologies to identify and measure the properties of a wide range of particles. The complex magnetic field and big amount of material within the ATLAS detector, along with the high collision rate of the accelerator, make track reconstruction very challenging. Track parameter and the associated error propagation is at the heart of almost any reconstruction algorithm, hence good accuracy and high speed — along with the consideration of material effects, such as energy loss and multiple scattering — are essential to the ATLAS tracking algorithms, such as the *simultaneous track and error propagation* (STEP) algorithm presented here. This algorithm transports the track parameters and associated covariance matrices through the dense volumes of the simplified ATLAS material description — the so-called tracking geometry [2] — which approximates the material distribution of the ATLAS calorimeter and muon spectrometer by a set of blended dense volumes to speed up the tracking process. This paper describes the error propagation of the STEP algorithm, while the transport of the track parameters is found in ref. [3].

Error propagation is usually handled in a purely analytical or numerical way. The first case is possible when the track model is explicitly given, thus allowing a direct derivation from the track model of the Jacobian needed to transport the covariance matrix. Unfortunately, explicit track models are limited to straight lines or helices, only useful in a vanishing or homogeneous magnetic field. Within the inhomogeneous ATLAS magnetic field, a numerical approach is necessary. The simplest numerical way of finding the derivatives of the transport Jacobian involves the propagation of one auxiliary track for every track parameter, which usually amounts to five additional tracks.

There is, however, a third alternative to the error propagation; the semi-analytical *Bugge-Myrheim method* [4]. This method propagates the transport Jacobian in parallel with the track parameters at little extra cost. Although this method has been known for many years, its accuracy and speed are not well documented in the scientific literature. In this paper we study the quality and speed of the Bugge-Myrheim method as a function of the accuracy of the underlying track parameter propagation. Furthermore, we look at the impact of the magnetic field and energy loss gradients on the accuracy and speed of the error propagation. We also show that the Bugge-Myrheim method is significantly faster than any purely numerical approach.

In section 2 we describe the error propagation in general before going into detail on the Bugge-Myrheim method in section 3. Furthermore, we look at the numerical error propagation — used for validating the semi-analytical error propagation — in section 4. In section 5 we compare the elements of the transport Jacobian obtained by the semi-analytical and the numerical error propagation. Moreover, in section 6 we perform a statistical test of the semi-analytically transported covariance matrix. Finally, we present a short conclusion in section 7.

Natural units ($\hbar = c = 1$) are used throughout this paper, and vectors and matrices are generally given in bold italic and bold capital letters, respectively.

2 Error propagation

The track parameters are often reconstructed from empirical data with associated uncertainties introduced by noise from the material interactions during the parameter transport, and uncertainties related to the misalignment and limited resolution of the detector. Here we focus on transporting the intrinsic measurement errors arising from the limited detector resolution, figure 1, leaving the discussion of the noise contributions from the material interactions to ref. [5]. It should, however, be noted that the noise contributions are added to the measurement errors during the propagation and hence are transported in the exact same way.

The measurement errors are given by the symmetric 5×5 covariance matrix with entry ij;

$$\Sigma_{ij} = \langle(\xi_i - \langle\xi_i\rangle)(\xi_j - \langle\xi_j\rangle)\rangle \tag{2.1}$$

where $\boldsymbol{\xi}$ is a vector of the *local track parameters*

$$\boldsymbol{\xi} = \begin{bmatrix} l_0 \\ l_1 \\ \phi \\ \theta \\ \lambda \end{bmatrix} \tag{2.2}$$

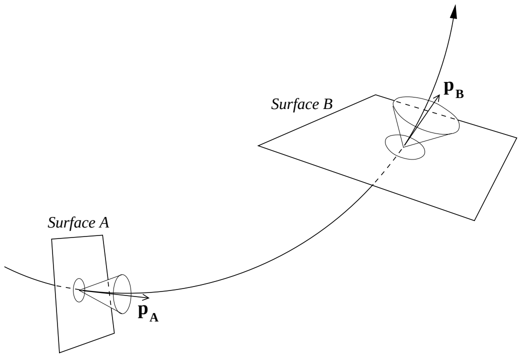

Figure 1. Transporting the track parameters and the associated uncertainties (covariance matrix) — indicated by the ellipses and cones — from one surface to another. The uncertainty of the momentum magnitude is omitted in this illustration.

and $\langle \xi_i \rangle$ are the expectation values of these parameters. The local track parameters [6] are defined by the local track position at a surface (l_0, l_1), the direction of the track momentum in the global ATLAS coordinate system (ϕ, θ) and the signed inverse of the momentum $(\lambda \equiv q/p)$. The global Cartesian, right-handed ATLAS coordinate system is approximately given by the LHC tunnel centre (x), the earth's surface (y) and the LHC beam pipe (z). The spherical polar coordinates ϕ and θ are defined as follows within this coordinate system; the azimuthal angle ϕ is given by the opening between the projection of the momentum into the x-y plane, and the x-axis, while the polar angle θ is given by the opening between the momentum and the z-axis. Together they define the direction of the momentum in the ATLAS coordinate system unambiguously, giving the following relations between the momentum components (p_x, p_y, p_z) and the angles (ϕ, θ);

$$\begin{aligned} p_x &= p\cos\phi\sin\theta \\ p_y &= p\sin\phi\sin\theta \\ p_z &= p\cos\theta \end{aligned} \tag{2.3}$$

The *global track parameters* used within STEP — to accommodate the propagation of bending tracks — are given in the above-mentioned global ATLAS coordinate system by

$$(x, y, z, T^x, T^y, T^z, \lambda)$$

where $\boldsymbol{T} = \boldsymbol{p}/p$ is the normalized tangent vector to the track, and (x, y, z) is the track position.

If the track model is explicitly given, the common approach to transporting the covariance matrix is to expand the analytical parameter propagation functions to first order in a Taylor series, and use these derivatives to propagate the covariance matrix in an approximate way. This is called *linear error propagation*. The availability of the derivatives of the propagated track parameters with respect to those at the starting point of the propagation — the so-called *Jacobian* **J** — is therefore essential to the linear error propagation. In our case, the Jacobian becomes a 5×5 matrix

$$\mathbf{J} = \begin{bmatrix} \frac{\partial l_0^{\text{final}}}{\partial l_0^{\text{initial}}} & \cdots & \frac{\partial l_0^{\text{final}}}{\partial \lambda^{\text{initial}}} \\ \vdots & \ddots & \vdots \\ \frac{\partial \lambda^{\text{final}}}{\partial l_0^{\text{initial}}} & \cdots & \frac{\partial \lambda^{\text{final}}}{\partial \lambda^{\text{initial}}} \end{bmatrix} \tag{2.4}$$

and transporting the symmetric covariance matrix by linear error propagation simply becomes a similarity transformation

$$\boldsymbol{\Sigma}_{\text{final}} = \mathbf{J} \cdot \boldsymbol{\Sigma}_{\text{initial}} \cdot \mathbf{J}^{\mathrm{T}} \tag{2.5}$$

However, because of the inhomogeneous magnetic field of ATLAS, and the resulting lack of explicit analytical functions for the propagation of the track parameters, the Jacobian cannot be calculated directly. The linear error propagation (2.5) is still valid, but the Jacobian must be obtained in another way. This is done in three steps; first, we find the Jacobian required for transforming the covariance matrix from the initial local track parameters to the initial global track parameters used within the STEP algorithm. Furthermore, this Jacobian is transported along with the track parameters to the destination surface. Finally, it is multiplied by the Jacobian which transforms the covariance matrix to the local track parameters at the destination surface. The resulting Jacobian (2.4) transports the covariance matrix from one set of local track parameters at the initial surface to another set of local track parameters at the destination surface (2.5). The initial and destination surfaces can be picked independently from any of the five surfaces defined within the ATLAS event data model, and positioned arbitrarily. In this paper we focus only on the second of the three steps; the transport of the Jacobian along the track.

3 Semi-analytical error propagation by using the Bugge-Myrheim method

As mentioned above, the common way of obtaining the derivatives (Jacobian) needed for the linear error propagation is to expand the parameter propagation functions to first order in a Taylor series. The lack of analytical parameter propagation functions in this case unfortunately makes this approach impossible. Another common technique is the numerical error propagation described in section 4. This method is slow, but robust and accurate, making it useful for testing the error propagation. A third way of obtaining the Jacobian is to differentiate the recursion formulae of the numerical integration method directly. This is the essence of the Bugge-Myrheim method. For reasons of efficiency and consistency, the natural choice is to pick the same integration method as used in the STEP parameter propagation, which is the adaptive Runge-Kutta-Nyström method. The Bugge-Myrheim method, however, follows the same principles regardless of the chosen integration method, only the recursion formulae change.

The basic idea of the adaptive Runge-Kutta-Nyström method is to divide the integration interval into steps and solve each step 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 h to make the procedure adaptive. The solution of every step is estimated by evaluating the equation of motion u'' at four different points — 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.

In the parameter propagation, we find the propagated global track parameters — where Λ is the integrated change in λ, or the total energy loss — of the equation of motion

$$\boldsymbol{u} = \begin{bmatrix} x \\ y \\ z \\ \Lambda \end{bmatrix}, \qquad \boldsymbol{u}' = \frac{d\boldsymbol{u}}{ds} = \begin{bmatrix} T^x \\ T^y \\ T^z \\ \lambda \end{bmatrix} \tag{3.1}$$

by integrating their respective differential equations by using some recursion formulae F and G. Given the Runge-Kutta-Nyström method, one step (numbered by n) becomes

$$\begin{aligned} \boldsymbol{u}_{n+1} &= F(s_n, \boldsymbol{u}_n, \boldsymbol{u}'_n) = \boldsymbol{F}_n(\boldsymbol{u}_n, \boldsymbol{u}'_n) = \boldsymbol{u}_n + h\boldsymbol{u}'_n + \frac{h^2}{6}(\boldsymbol{u}''_1 + \boldsymbol{u}''_2 + \boldsymbol{u}''_3) \\ \boldsymbol{u}'_{n+1} &= G(s_n, \boldsymbol{u}_n, \boldsymbol{u}'_n) = \boldsymbol{G}_n(\boldsymbol{u}_n, \boldsymbol{u}'_n) = \boldsymbol{u}'_n + \frac{h}{6}(\boldsymbol{u}''_1 + 2\boldsymbol{u}''_2 + 2\boldsymbol{u}''_3 + \boldsymbol{u}''_4) \end{aligned} \tag{3.2}$$

To obtain the derivatives (Jacobian) of the propagated global track parameters with respect to the initial local track parameters (i denoting initial)

$$\boldsymbol{\xi}^i = \begin{bmatrix} l_0^i \\ l_1^i \\ \phi^i \\ \theta^i \\ \lambda^i \end{bmatrix} \tag{3.3}$$

the recursion formulae (3.2) have to be differentiated with respect to $\boldsymbol{\xi}^i$, giving

$$\mathbf{J}_{n+1} = \begin{bmatrix} \frac{\partial \boldsymbol{u}_{n+1}}{\partial \boldsymbol{\xi}^i} \\ \frac{\partial \boldsymbol{u}'_{n+1}}{\partial \boldsymbol{\xi}^i} \end{bmatrix} = \begin{bmatrix} \frac{\partial \boldsymbol{F}_n}{\partial \boldsymbol{\xi}^i} \\ \frac{\partial \boldsymbol{G}_n}{\partial \boldsymbol{\xi}^i} \end{bmatrix} = \begin{bmatrix} \frac{\partial \boldsymbol{F}_n}{\partial \boldsymbol{u}_n} & \frac{\partial \boldsymbol{F}_n}{\partial \boldsymbol{u}'_n} \\ \frac{\partial \boldsymbol{G}_n}{\partial \boldsymbol{u}_n} & \frac{\partial \boldsymbol{G}_n}{\partial \boldsymbol{u}'_n} \end{bmatrix} \cdot \begin{bmatrix} \frac{\partial \boldsymbol{u}_n}{\partial \boldsymbol{\xi}^i} \\ \frac{\partial \boldsymbol{u}'_n}{\partial \boldsymbol{\xi}^i} \end{bmatrix} = \mathbf{D}_n \cdot \mathbf{J}_n \tag{3.4}$$

where the derivatives $\partial \boldsymbol{u}_n / \partial \boldsymbol{\xi}^i$ and $\partial \boldsymbol{u}'_n / \partial \boldsymbol{\xi}^i$ of the 8×5 Jacobian $\mathbf{J}$ are given by the 4×5 matrices

$$\frac{\partial \boldsymbol{u}_n}{\partial \boldsymbol{\xi}^i} = \begin{bmatrix} \frac{\partial x_n}{\partial l_0^i} & \cdots & \frac{\partial x_n}{\partial \lambda^i} \\ \vdots & \ddots & \vdots \\ \frac{\partial \Lambda_n}{\partial l_0^i} & \cdots & \frac{\partial \Lambda_n}{\partial \lambda^i} \end{bmatrix}, \quad \frac{\partial \boldsymbol{u}'_n}{\partial \boldsymbol{\xi}^i} = \begin{bmatrix} \frac{\partial T_n^x}{\partial l_0^i} & \cdots & \frac{\partial T_n^x}{\partial \lambda^i} \\ \vdots & \ddots & \vdots \\ \frac{\partial \lambda_n}{\partial l_0^i} & \cdots & \frac{\partial \lambda_n}{\partial \lambda^i} \end{bmatrix} \tag{3.5}$$

$\mathbf{D}_n$ is an 8×8 matrix containing the recursion formulae $\boldsymbol{F}_n$ and $\boldsymbol{G}_n$ differentiated with respect to the global track parameters

$$\mathbf{D}_n = \frac{\partial(\boldsymbol{F}_n, \boldsymbol{G}_n)}{\partial(\boldsymbol{u}_n, \boldsymbol{u}'_n)} = \begin{bmatrix} \frac{\partial \boldsymbol{F}_n}{\partial \boldsymbol{u}_n} & \frac{\partial \boldsymbol{F}_n}{\partial \boldsymbol{u}'_n} \\ \frac{\partial \boldsymbol{G}_n}{\partial \boldsymbol{u}_n} & \frac{\partial \boldsymbol{G}_n}{\partial \boldsymbol{u}'_n} \end{bmatrix} \tag{3.6}$$

giving the 4×4 matrices

$$\frac{\partial \boldsymbol{F}_n}{\partial \boldsymbol{u}_n} = \begin{bmatrix} \frac{\partial F_n^x}{\partial x_n} & \cdots & \frac{\partial F_n^x}{\partial \Lambda_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial F_n^\Lambda}{\partial x_n} & \cdots & \frac{\partial F_n^\Lambda}{\partial \Lambda_n} \end{bmatrix}, \quad \frac{\partial \boldsymbol{F}_n}{\partial \boldsymbol{u}'_n} = \begin{bmatrix} \frac{\partial F_n^x}{\partial T_n^x} & \cdots & \frac{\partial F_n^x}{\partial \lambda_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial F_n^\Lambda}{\partial T_n^x} & \cdots & \frac{\partial F_n^\Lambda}{\partial \lambda_n} \end{bmatrix} \tag{3.7}$$

and

$$\frac{\partial \boldsymbol{G}_n}{\partial \boldsymbol{u}_n} = \begin{bmatrix} \frac{\partial G_n^{T^x}}{\partial x_n} & \cdots & \frac{\partial G_n^{T^x}}{\partial \Lambda_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial G_n^{\lambda}}{\partial x_n} & \cdots & \frac{\partial G_n^{\lambda}}{\partial \Lambda_n} \end{bmatrix}, \quad \frac{\partial \boldsymbol{G}_n}{\partial \boldsymbol{u}'_n} = \begin{bmatrix} \frac{\partial G_n^{T^x}}{\partial T_n^x} & \cdots & \frac{\partial G_n^{T^x}}{\partial \lambda_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial G_n^{\lambda}}{\partial T_n^x} & \cdots & \frac{\partial G_n^{\lambda}}{\partial \lambda_n} \end{bmatrix} \tag{3.8}$$

By writing the recursion formulae of the derivatives as a product of $\mathbf{D}_n$ and $\mathbf{J}_n$ (3.4), we can differentiate the recursion formulae $\boldsymbol{F}_n$ and $\boldsymbol{G}_n$ with respect to the global track parameters $\boldsymbol{u}_n$ and $\boldsymbol{u}'_n$ instead of the initial local track parameters $\boldsymbol{\xi}^i$. This simplifies the differentiation a lot, giving

$$\begin{aligned} \frac{\partial \boldsymbol{F}_n}{\partial \boldsymbol{u}_n} &= 1 + \frac{h^2}{6}\left(\frac{\partial \boldsymbol{u}''_1}{\partial \boldsymbol{u}_n} + \frac{\partial \boldsymbol{u}''_2}{\partial \boldsymbol{u}_n} + \frac{\partial \boldsymbol{u}''_3}{\partial \boldsymbol{u}_n}\right) \\ \frac{\partial \boldsymbol{F}_n}{\partial \boldsymbol{u}'_n} &= h + \frac{h^2}{6}\left(\frac{\partial \boldsymbol{u}''_1}{\partial \boldsymbol{u}'_n} + \frac{\partial \boldsymbol{u}''_2}{\partial \boldsymbol{u}'_n} + \frac{\partial \boldsymbol{u}''_3}{\partial \boldsymbol{u}'_n}\right) \\ \frac{\partial \boldsymbol{G}_n}{\partial \boldsymbol{u}_n} &= \frac{h}{6}\left(\frac{\partial \boldsymbol{u}''_1}{\partial \boldsymbol{u}_n} + 2\frac{\partial \boldsymbol{u}''_2}{\partial \boldsymbol{u}_n} + 2\frac{\partial \boldsymbol{u}''_3}{\partial \boldsymbol{u}_n} + \frac{\partial \boldsymbol{u}''_4}{\partial \boldsymbol{u}_n}\right) \\ \frac{\partial \boldsymbol{G}_n}{\partial \boldsymbol{u}'_n} &= 1 + \frac{h}{6}\left(\frac{\partial \boldsymbol{u}''_1}{\partial \boldsymbol{u}'_n} + 2\frac{\partial \boldsymbol{u}''_2}{\partial \boldsymbol{u}'_n} + 2\frac{\partial \boldsymbol{u}''_3}{\partial \boldsymbol{u}'_n} + \frac{\partial \boldsymbol{u}''_4}{\partial \boldsymbol{u}'_n}\right) \end{aligned} \tag{3.9}$$

To calculate these derivatives explicitly, we need to differentiate the individual stages of the Runge-Kutta-Nyström method with respect to the global track parameters,

$$\mathbf{A}_k = \frac{\partial \boldsymbol{u}''_k}{\partial \boldsymbol{u}'_n}, \qquad \mathbf{C}_k = \frac{\partial \boldsymbol{u}''_k}{\partial \boldsymbol{u}_n} \tag{3.10}$$

where k denotes the individual stages, and $\boldsymbol{u}''_k$ is given — in a general form — by the equations of motion of the global track parameters [4]

$$\begin{aligned} x'' &= \lambda(T^y B_z - T^z B_y) \\ y'' &= \lambda(T^z B_x - T^x B_z) \\ z'' &= \lambda(T^x B_y - T^y B_x) \\ \Lambda'' &= -\frac{\lambda^3 g E}{q^2} \end{aligned} \tag{3.11}$$

The last equation handles the energy loss, with E being the energy and g the energy loss per unit distance. The energy loss and its gradient varies little within each recursion step, hence the values calculated in the first stage are recycled by the following stages. This lowers the computing cost considerably.

Writing the 4×4 $\mathbf{A}_k$ and $\mathbf{C}_k$ matrices in a general form, we get

$$\mathbf{A} = \begin{bmatrix} \frac{\partial x''}{\partial T^x} & \cdots & \frac{\partial x''}{\partial \lambda} \\ \vdots & \ddots & \vdots \\ \frac{\partial \Lambda''}{\partial T^x} & \cdots & \frac{\partial \Lambda''}{\partial \lambda} \end{bmatrix} = \begin{bmatrix} 0 & \lambda B_z & -\lambda B_y & T^y B_z - T^z B_y \\ -\lambda B_z & 0 & \lambda B_x & T^z B_x - T^x B_z \\ \lambda B_y & -\lambda B_x & 0 & T^x B_y - T^y B_x \\ 0 & 0 & 0 & \left(\frac{1}{\lambda}(3 - \frac{p^2}{E^2}) + \frac{1}{g}\frac{\partial g}{\partial \lambda}\right)\Lambda'' \end{bmatrix} \tag{3.12}$$

and

$$\mathbf{C} = \begin{bmatrix} \frac{\partial x''}{\partial x} & \cdots & \frac{\partial x''}{\partial \Lambda} \\ \vdots & \ddots & \vdots \\ \frac{\partial \Lambda''}{\partial x} & \cdots & \frac{\partial \Lambda''}{\partial \Lambda} \end{bmatrix} = \begin{bmatrix} \lambda(T^y \frac{\partial B_z}{\partial x} - T^z \frac{\partial B_y}{\partial x}) & \lambda(T^y \frac{\partial B_z}{\partial y} - T^z \frac{\partial B_y}{\partial y}) & \lambda(T^y \frac{\partial B_z}{\partial z} - T^z \frac{\partial B_y}{\partial z}) & 0 \\ \lambda(T^z \frac{\partial B_x}{\partial x} - T^x \frac{\partial B_z}{\partial x}) & \lambda(T^z \frac{\partial B_x}{\partial y} - T^x \frac{\partial B_z}{\partial y}) & \lambda(T^z \frac{\partial B_x}{\partial z} - T^x \frac{\partial B_z}{\partial z}) & 0 \\ \lambda(T^x \frac{\partial B_y}{\partial x} - T^y \frac{\partial B_x}{\partial x}) & \lambda(T^x \frac{\partial B_y}{\partial y} - T^y \frac{\partial B_x}{\partial y}) & \lambda(T^x \frac{\partial B_y}{\partial z} - T^y \frac{\partial B_x}{\partial z}) & 0 \\ -\frac{\lambda^3 E}{q^2} \frac{\partial g}{\partial x} & -\frac{\lambda^3 E}{q^2} \frac{\partial g}{\partial y} & -\frac{\lambda^3 E}{q^2} \frac{\partial g}{\partial z} & 0 \end{bmatrix} \tag{3.13}$$

With the help of these matrices, we find the elements of $\mathbf{D}_n$, which is multiplied by $\mathbf{J}_n$ to produce the transported Jacobian $\mathbf{J}_{n+1}$ (3.4). This procedure is repeated for every recursion step, transforming **J** along the way.

When applied to real problems, the gradients of **A** and **C** are usually quite costly to calculate, hence it is common practice to set all of the $\partial g/\partial\lambda$, $\partial B_i/\partial x_j$ and $\partial g/\partial x_j$ gradients, i and j indicating the x, y and z components, to zero. This is, however, only correct for the material gradients $\partial g/\partial x_j$ of the blended dense volumes of the simplified ATLAS material description.

4 Numerical error propagation

To test the semi-analytical error propagation, we need an alternative way of calculating the derivatives of the Jacobian (2.4). The most straightforward way is by using the definition of the numerical derivative

$$f'(\xi_i) \approx \frac{f(\xi_i + h_i) - f(\xi_i)}{h_i} \tag{4.1}$$

where $f(\xi_i)$ propagates the local track parameters — denoted by i — from the initial surface to the target surface, while h_i is kept sufficiently small, ideally zero. By using the above definition of the derivative, we vary the initial local track parameters by a small amount h_i, one at a time. This is the key to knowing exactly how these variations translate to the final local track parameters. Registering the changes to the final parameters gives us the 25 derivatives of the Jacobian.

Though very easy and straightforward, this method is quite inaccurate. One way of increasing the accuracy is by using the symmetric derivative

$$g(h_i) \approx \frac{f(\xi_i + h_i) - f(\xi_i - h_i)}{2h_i} \tag{4.2}$$

which typically has a fractional error two orders of magnitude better than the original definition of the derivative [7].

To further increase the accuracy, we use a numerical method called Ridders' algorithm [7]. The essence of Ridders' algorithm is to parameterize the symmetric derivative as a function of h_i alone by calculating it for descending values of h_i, figure 2. This parameterization of $g(h_i)$ is used to estimate the derivative in the limit $h_i \rightarrow 0$. Since it has to be done for every derivative, this method is very time consuming and only useful for testing.

Compared to the semi-analytical error propagation even the simplest numerical error propagation is slow, needing at least five additional parameter propagations for every track, increasing the computing time accordingly.

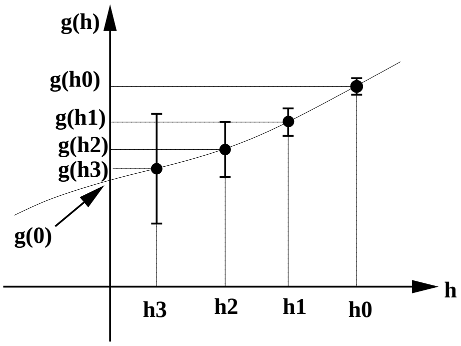

Figure 2. Schematic plot of the Ridders algorithm. The graph shows the parameterization of the symmetric derivative $g(h_i)$. This algorithm reduces the problem with the increasing rounding errors — indicated by the error bars — of the symmetric derivative when $h_i \to 0$.

5 Validating the Jacobian in an inhomogeneous magnetic field including energy loss

To get a complete understanding of the semi-analytical error propagation, we need to study the Jacobian terms in a realistic, inhomogeneous magnetic field with energy loss. The test setup involves propagating muons — through solid Silicon in the realistic ATLAS magnetic field — 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 interaction point of the ATLAS detector. The particles are propagated towards a target surface randomly placed and rotated in a cube with sides of 20 m centered in the detector. Generally, muons experience a 5 GeV energy loss in the ATLAS detector — regardless of their initial momentum — which is doubled by the solid material of the test volume. The most energetic muons of the track sample are displaced by a few cm from a straight line by the ATLAS magnetic field on their 15–20 m trip through the detector, while the intermediate muons at around 100 GeV are displaced 10–20 cm and the lowest energy muons might be shifted several meters.

During this test, the derivatives required by the error propagation are calculated twice; first semi-analytically by the Bugge-Myrheim method, and then numerically by the Ridders algorithm. The numerical derivatives define the baseline for the semi-analytical terms. To assure the quality of the numerical derivatives, the STEP propagator at an error tolerance of 10^{-8} is used for calculating the symmetric derivatives of the Ridders algorithm. The error tolerance is a user specified number steering the accuracy of the propagation, a low error tolerance giving a high accuracy, and vice versa. The *absolute*, *relative residual*

$$\frac{|\text{semi-analytical derivative} - \text{numerical derivative}|}{|\text{numerical derivative}|} \tag{5.1}$$

is then used to compare the 25 derivatives of the semi-analytical and numerical Jacobians.

Figure 3 shows three histograms of the logarithm of the absolute, relative residuals of the Jacobian element $\partial l_1^f/\partial\lambda^i$, f and i indicating the final and initial values. These histograms are typical of all the $\partial l_0^f/\partial\boldsymbol{\xi}^i$ and $\partial l_1^f/\partial\boldsymbol{\xi}^i$ Jacobian elements.

Figure 4 shows the effect on the residuals of two Jacobian terms by only including one type of gradient into the calculation of the semi-analytical derivatives. These terms are only sensitive

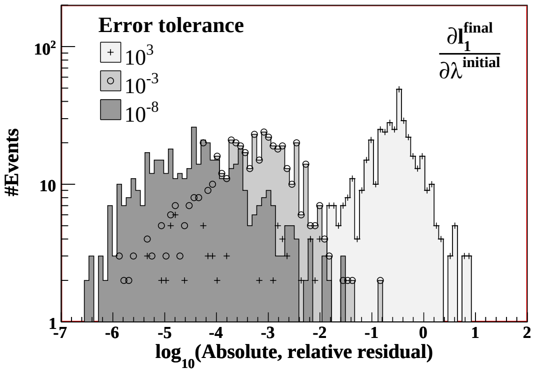

Figure 3. Logarithms of the absolute, relative residuals of the $\partial l_1^f/\partial\lambda^i$ Jacobian term in an inhomogeneous magnetic field with energy loss. The semi-analytical derivatives are calculated at three error tolerances with both gradients included, whereas the numerical derivatives are all calculated at an error tolerance of 10^{-8}.

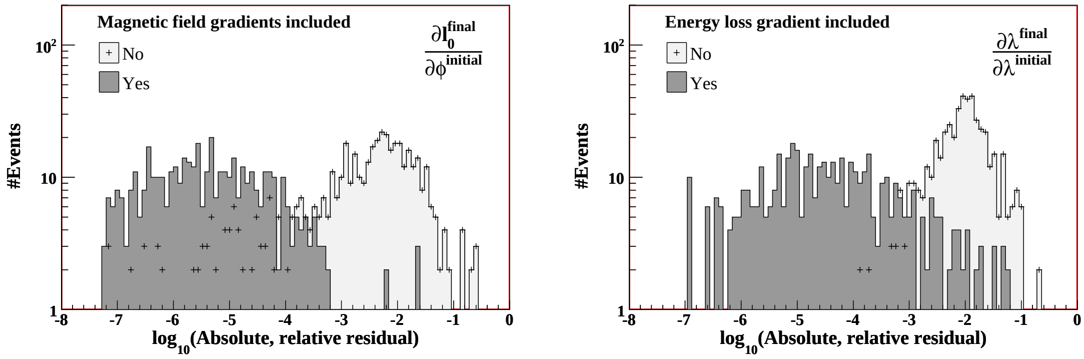

Figure 4. Logarithms of the absolute, relative residuals of two Jacobian terms in an inhomogeneous magnetic field with energy loss. The semi-analytical derivatives are calculated with, and without including the magnetic field gradients $\partial B_i/\partial x_j$ (left) and the energy loss gradient $\partial g/\partial\lambda$ (right). Both the semi-analytical and numerical derivatives are calculated at an error tolerance of 10^{-8}.

to either the magnetic field gradients or the energy loss gradient. Due to the underlying single precision of the analysis program used to produce the plots (ROOT [8]), no relative difference better than approximately 10^{-7} is seen. Entries with better relative precision become identically zero and are not shown.

Figure 5 shows the mean values of residuals of a selection of Jacobian terms with and without both gradients included. All of the semi-analytical derivatives are sensitive to the gradients, especially the angular derivatives.

The improvements in the residuals when turning on the magnetic field gradients are presented on the left-hand side of figure 6, while the additional improvements by including the energy loss gradient are shown to the right. The effect of the energy loss gradient is only seen in the last column of the Jacobian, $\partial\boldsymbol{\xi}^f/\partial\lambda^i$, illustrated by the constant $\partial l_0^f/\partial\phi^i$ residual in the right-hand plot of figure 6, whereas the effects of the magnetic field gradients show up all over the Jacobian, except in the last row $\partial\lambda^f/\partial\boldsymbol{\xi}^i$, as illustrated by the constant $\partial\lambda^f/\partial\lambda^i$ residual in the left-hand plot of the same figure.

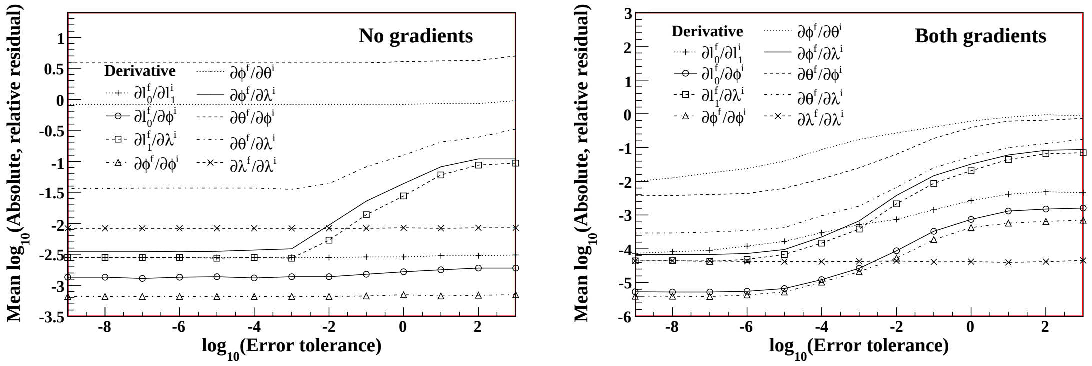

Figure 5. Mean values of the logarithms of the absolute, relative residuals in an inhomogeneous magnetic field with energy loss. The semi-analytical derivatives are calculated at different error tolerances by including no gradients (left) and both gradients (right), whereas the numerical derivatives are all found at an error tolerance of 10^{-8}.

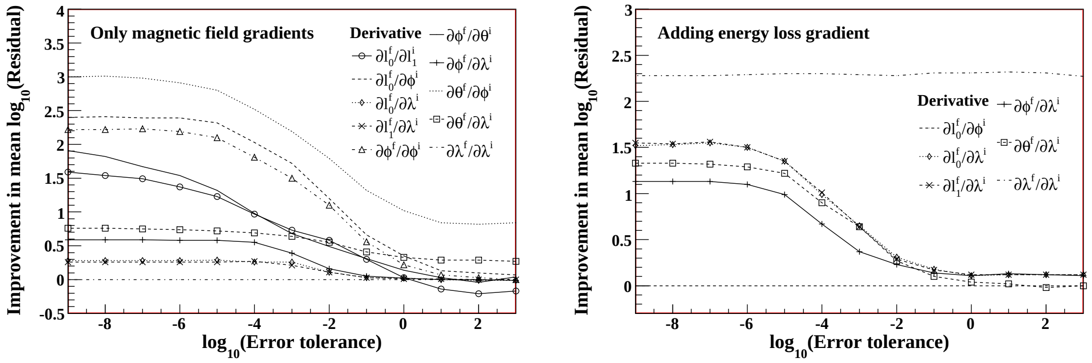

Figure 6. Improvements in the mean values of the logarithms of the absolute, relative residuals by including the magnetic field gradients (left), and the additional improvements by including the energy loss gradient (right) in an inhomogeneous magnetic field with energy loss. The semi-analytical derivatives are calculated at different error tolerances, whereas the numerical derivatives are all found at an error tolerance of 10^{-8}.

Figure 7 shows the additional computing time — relative to the STEP parameter propagation — spent by the semi-analytical error propagation, magnetic field and energy loss gradients. The error propagation is only done after the adaptive parameter propagation has found the optimal step length, making the nominal computing cost of the error propagation relatively stable over the whole error tolerance range. Thus, the drop in the additional computing cost of the error propagation at low error tolerances is mostly caused by the increased computing cost of the parameter propagation.

6 Verifying the propagated covariance matrix in an inhomogeneous magnetic field including energy loss

In the previous sections we have looked at the individual Jacobian elements to get a deeper understanding of the error propagation. Now, we examine the final covariance matrix produced by

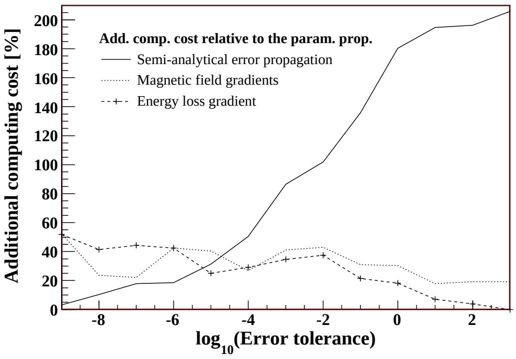

Figure 7. Additional computing time relative to the STEP parameter propagation.

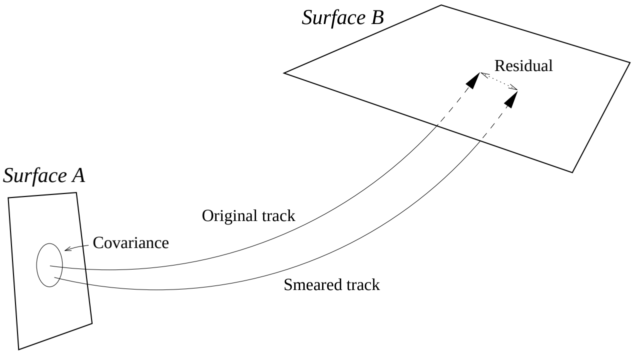

Figure 8. Testing the error propagation by smearing the initial track parameters according to the covariance matrix. The residual is normalized by the propagated covariance to produce the pulls, and multiplied by the inverse of the propagated covariance to find the chi-square. The angles and the momentum are smeared similarly to the positions shown in the figure.

the linear error propagation (2.5). From this transformation, we see that the elements of the final covariance matrix are sums and products of many initial covariance and Jacobian terms. Evaluating the final covariance elements on an individual basis becomes prohibitively difficult, yet testing the Jacobian alone is not sufficient to guarantee the quality of the error propagation. Only a full error propagation, using a realistic initial covariance matrix, allows us to test the significance of the missing, or inaccurate Jacobian elements, and the gradients. To perform this test, we use the fact that the initial covariance matrix defines the Gaussian variances and correlations of the initial track parameters. By varying the initial track parameters according to their associated covariance matrix before propagating them to the target surface, the variation of the final track parameters should be reflected in the final covariance matrix. In short, we use the initial covariance matrix for smearing the simulated tracks and the final track parameters for statistically testing the propagated covariance matrix, figure 8.

6.1 Smearing the initial track parameters according to the covariance matrix

To simulate the variances and correlations of the initial local track parameters, we first decompose the initial covariance matrix into two triangular matrices by using Cholesky's method [7]

$$\mathbf{\Sigma}_{\text{initial}} = \mathbf{L} \cdot \mathbf{L}^{\mathrm{T}} \tag{6.1}$$

This method is easy to use and sufficient for decomposing symmetric, positive definitive matrices such as the covariance matrix.

The elements of the initial covariance matrices are picked at random from Gaussian distributions with mean values of zero and widths of 50 μm for the positions l_0 and l_1, 1 mrad for the angles ϕ and θ, and 1% for the inverse momentum λ. These are realistic values of the resolution of the ATLAS detector, except from the 1% λ uncertainty, which is too optimistic. This is kept low due to the big amount of material — and hence large energy losses of the particles — in the test setup.

After decomposing the initial covariance matrix, we use it to smear [9] the initial local track parameters $\boldsymbol{\mu}^i$ (i denoting initial)

$$\boldsymbol{\xi}^i = \boldsymbol{\mu}^i + \mathbf{L} \cdot \boldsymbol{\eta} \tag{6.2}$$

where $\mathbf{L}$ is the lower triangular matrix obtained through the Cholesky decomposition (6.1), and $\boldsymbol{\eta}$ is a vector of five independent variables picked at random from a Gaussian distribution of mean zero and variance one. Equation (6.2) assumes that the initial local track parameters are Gaussian distributed and smears them accordingly by using the initial covariance matrix.

6.2 Statistical validation of the semi-analytically propagated covariance matrix

In this test, muon tracks are generated — with their initial parameters smeared according to the above procedure — and propagated by using the test setup described in section 5, replacing the Silicon with Iron. Enough tracks are generated to make the statistical uncertainties insignificant. The track parameter and error propagation is done by STEP with the magnetic field and energy loss gradients included, at an error tolerance of 10^{-8} to assure the quality of the tracks. Each track is propagated twice to produce the undisturbed $\boldsymbol{\mu}$ and smeared $\boldsymbol{\xi}$ local track parameters at the target surface. The $\boldsymbol{\xi} - \boldsymbol{\mu}$ residuals are then statistically compared to the semi-analytically propagated covariance matrix $\mathbf{\Sigma}_{\text{final}}$ by using the normalized residuals, or *pull values*

$$\hat{\xi}_k^j = \frac{\xi_k^j - \mu_k^j}{\sqrt{\Sigma_{\text{final},k}^{jj}}} \tag{6.3}$$

and the *chi-square*

$$\chi_k^2 = (\boldsymbol{\xi}_k - \boldsymbol{\mu}_k)^{\mathrm{T}} \cdot \mathbf{\Sigma}_{\text{final},k}^{-1} \cdot (\boldsymbol{\xi}_k - \boldsymbol{\mu}_k) \tag{6.4}$$

with k indicating the simulated tracks and j the track parameters.

Since ξ_k^j is Gaussian distributed around μ_k^j, the pull values should be Gaussian distributed around zero. Moreover, if the propagated covariance $\mathbf{\Sigma}_{\text{final},k}$ is correct, the width of the pull values should be normalized to one. All of the pull values presented in figure 9 satisfy these requirements, showing good agreement between the semi-analytical error propagation and the simulation. The

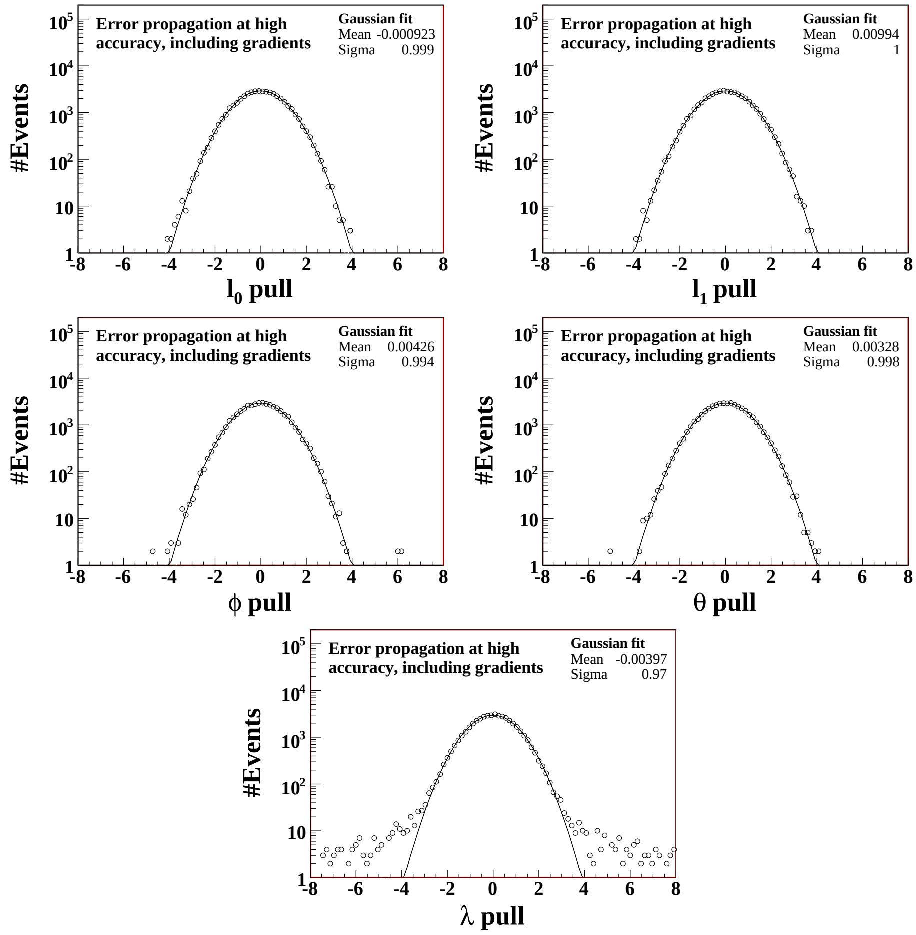

Figure 9. Pull values of the final parameters of tracks smeared by the initial covariance matrix, and their Gaussian fits (solid lines). The covariance matrices are propagated semi-analytically at an error tolerance of 10^{-8} with the gradients included.

tails of the λ pull are intrinsic to the semi-analytical error propagation and arise from the information loss caused by introducing the temporary global track parameters during the error propagation.

Whereas the pull values are calculated for each parameter of the simulated track, the chi-square incorporates the whole covariance matrix and all of the track parameters. Assuming that all five track parameters are Gaussian distributed, and that these distributions obey the variances and correlations given by covariance matrix, the test chi-square distribution should be similar to the standard chi-square distribution corresponding to five degrees of freedom. By integrating the standard chi-square distribution from the test chi-square to infinity, we get the so-called *p-value*, or probability value of this test statistic. If the test chi-square distribution is correct, the p-value plot is flat. Inverting the covariance matrix by using singular value decomposition [7], we get the flat p-value plots of figure 10, showing good agreement between the semi-analytical error propagation and the simulation.

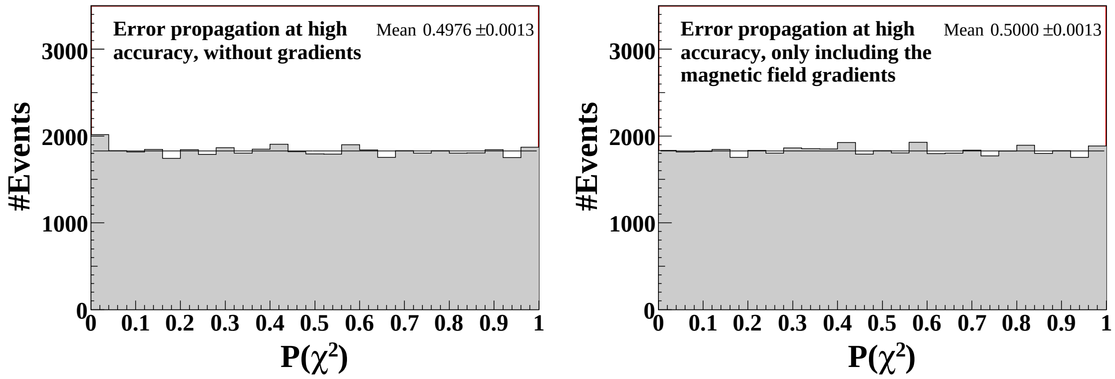

Figure 10. P-values of the final parameters of tracks smeared by the initial covariance matrix, and their linear fits. The covariance matrices are propagated semi-analytically at an error tolerance of 10^{-8}, excluding all of the the gradients (left) and by only including the magnetic field gradients (right).

6.3 Estimating the impact of the gradients on the semi-analytical error propagation

The covariance matrices of the pulls of figure 9 are all propagated by including the $\partial g/\partial\lambda$ and $\partial B_i/\partial x_j$ gradients discussed in section 5 into the error propagation. Including these gradients improves some elements of the Jacobian significantly. Such improvements are also seen when comparing the p-values found by only including the magnetic field gradients (right) to those found by excluding all of the gradients (left) in figure 10. The flat p-value plot found by only including the magnetic field gradients leaves little room for further improvement. Thus, only the magnetic field gradients — and not the energy loss gradient — are included into STEP by default. The gradients' influence on the pulls is insignificant, consequently they are not presented here.

Pull and p-values obtained by using an error tolerance of 10^{-2} for the semi-analytical error propagation — instead of the 10^{-8} used in figures 9 and 10 — produce similar plots, indicating little sensitivity to the error tolerance in the semi-analytical error propagation.

7 Conclusion

In this paper we have performed an extensive study of the Bugge-Myrheim method, gaining a quantitative understanding of the impact of the magnetic field and energy loss gradients on the accuracy and speed of the semi-analytical error propagation. Results show that only the magnetic field gradients have a visible effect on the covariance matrices transported by the semi-analytical error propagation in the ATLAS magnetic field, hence the energy loss gradient is left out of the error propagation by default.

The computing cost increase — relative to the parameter propagation — by adding the semi-analytical error propagation is less than 100% at medium and high accuracies. This is significantly less than the minimal computing cost increase of 500% seen in the numerical error propagation methods. Furthermore, the additional computing cost for including the magnetic field and energy loss gradients is around 30–40% for each type of gradient. Finally, the nominal computing cost and accuracy of the semi-analytical error propagation is relatively stable over the whole error tolerance range.

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] A. Salzburger et al., *The ATLAS Tracking Geometry Description*, ATLAS Public Note ATL-SOFT-PUB-2007-004 (2007).

[3] E. Lund et al., *Track parameter propagation through the application of a new adaptive Runge-Kutta-Nyström method in the ATLAS experiment*, ATLAS Public Note ATL-SOFT-PUB-2009-001, submitted to JINST (2009).

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

[5] E. Lund et al., *Treatment of energy loss and multiple scattering in the context of track parameter and covariance matrix propagation in continuous material in the ATLAS experiment*, ATLAS Public Note ATL-SOFT-PUB-2008-003 (2008).

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

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

[8] R. Brun and F. Rademakers, *ROOT: an object oriented data analysis framework*, *Nucl. Instrum. Meth.* **A 389** (1997) 81.

[9] PARTICLE DATA GROUP collaboration, C. Amsler et al., *Color Figures*, *Phys. Lett.* **B 667** (2008) 1340.