Analytical and Numerical Modelling of Particles in Electromagnetic Fields

John F C Paterson

April 19, 2024

I certify that this project report has been written by me, is a record of work carried out by me, and is essentially different from work undertaken for any other purpose or assessment.

Abstract

Electromagnetic fields apply a force on charged particles. Different configurations of fields will apply different forces, resulting in a wide variety of individual particle dynamics. In certain configurations of fields, it is possible to derive a full analytical solution for the motion of a charged particle. This project will derive the full analytical solution for the position of a charged particle in two configurations of electromagnetic fields; a constant magnetic field and a combination of a constant electric field and a constant magnetic field. Furthermore, a perturbation solution was derived for an inhomogeneous magnetic field where the inhomogeneity is small. However, more often than not, these solutions are not possible to derive. Therefore numerical methods must be considered to derive the trajectory of a charged particle in an electromagnetic field. Hence, two numerical solvers were developed specifically, using normalised values, to model these trajectories. These were tested extensively against analytical solutions and each time they achieved the expected rate of convergence. These methods were used to find the trajectories of charged particles in various configurations of electromagnetic fields. Where possible, an error analysis was conducted against the analytical or perturbation solution. The numerical methods developed are capable of modelling trajectories in far more complex fields than have been considered in this project, therefore these more complex fields will be a focus of my future work in this area. Overall, this project aims to accurately model the individual dynamics of charged particles in electromagnetic fields.

Keywords: Particle orbits, electromagnetic fields, analytical modelling, numerical modelling, gradient drift, curvature drift.

Contents

1 Introduction

In this project, I will be considering the motion of individual charged particles in various electromagnetic (EM) fields. The equation of motion for these particles is given by the non-relativistic Lorentz Equation. The equation will be solved both numerically and analytically for increasingly complex EM fields, followed by an in-depth error analysis. The majority of the analytical side of this project will follow Section 2 from the textbook "The Physics of Plasmas" by Boyd and Sanderson [2003] and Professor Duncan Mackay's work in the field. The numerical work and code used in this project are solely the work of the author. Lastly, the applications of this topic, assumptions made, and an overview of the Lorentz Equation will be outlined in this section.

1.1 Lorentz Equation

The motion of charged particles in electromagnetic fields is governed by the Lorentz Equation, which is given by:

$$\mathbf{F}(\mathbf{r},t) = q\left(\mathbf{E}(\mathbf{r},t) + \mathbf{v}(\mathbf{r},t) \times \mathbf{B}(\mathbf{r},t)\right), \tag{1.1.1}$$

where $\mathbf{r} = (x(t), y(t), z(t))$ is the position of the particle, t is the time, q and m are the charge and the mass of the particle respectively, $\mathbf{v}$ is the velocity of the particle at given time and position, $\mathbf{B}$ and $\mathbf{E}$ are, respectively, the magnetic and electric fields the charged particle experiences at a given time and position, and $\mathbf{F}$ is the force exerted on the charged particle by the EM field at a given time and position.

When examining equation (1.1.1) it can be seen that it is the sum of force on the charged particle due to the magnetic and electric fields, respectively. Hence,

$$\begin{aligned} \mathbf{F}_E &= q\mathbf{E} \\ \mathbf{F}_B &= q\left(\mathbf{v} \times \mathbf{B}\right). \end{aligned}$$

In order to consider the particle motion we use Newton's Second Law, $\mathbf{F} = m\mathbf{a} = m(\ddot{x}, \ddot{y}, \ddot{z})$, where $\ddot{x}$ represents the second time derivative of the particle's position in the $\hat{\mathbf{x}}$-direction (ie acceleration in x). Therefore, using Newton's Second Law, equation (1.1.1) becomes:

$$\mathbf{a}(\mathbf{r},t) = \ddot{\mathbf{r}}(\mathbf{r},t) = \frac{q}{m}\left(\mathbf{E}(\mathbf{r},t) + \mathbf{v}(\mathbf{r},t) \times \mathbf{B}(\mathbf{r},t)\right). \tag{1.1.2}$$

Evaluating the cross product and splitting it into components will result in the three coupled 2nd order ordinary differential equations (ODEs) shown below;

$$\ddot{x} = \frac{q}{m}\left(E_x + \dot{y}B_z - \dot{z}B_y\right) \tag{1.1.3}$$

$$\ddot{y} = \frac{q}{m}\left(E_y + \dot{z}B_x - \dot{x}B_z\right) \tag{1.1.4}$$

$$\ddot{z} = \frac{q}{m}\left(E_z + \dot{x}B_y - \dot{y}B_x\right), \tag{1.1.5}$$

where E_x is the x component of the electric field, B_x is the x component of the magnetic field and again using the dot notation to depict time derivatives.

This system of ODEs is what the remainder of the project will focus on solving in various ways, both analytically and numerically, for a variety of combinations of EM fields.

1.2 Assumptions

Before reading the main body of work, it is important to consider the primary assumptions that have been made for the equations and techniques used to be valid. There will be further assumptions made throughout the paper. However, these are more specific and will be highlighted in the relevant section.

The main assumption of the Lorentz Equation (1.1.1) is that it only holds for non-relativistic velocities $v \ll c$, for speed of light c. This significantly simplifies the analysis of the equation as Special Relativity does not need to be taken into account, hence the equation remains valid without the need to adjust for the increase in mass or the contraction of lengths. Next, we must assume that the fields induced by the motion of the charged particle are negligible in comparison to the applied fields $\mathbf{B}$ and $\mathbf{E}$. Similarly, the Lorentz Equation assumes that the effect of self-consistent fields from neighbouring charges is negligible in comparison to the applied EM fields. In other words, the possible effects of any other particles are neglected. This is likely to be the case when dealing with a relatively strong $\mathbf{B}$ and $\mathbf{E}$ and having $v \ll c$. Moreover, the equation assumes that there are no collisions between particles. Finally, the applied EM fields must satisfy Maxwell's equations:

$$\begin{aligned} \nabla \times \mathbf{E} &= -\frac{\partial \mathbf{B}}{\partial t} \\ \nabla \cdot \mathbf{E} &= \frac{\rho}{\epsilon_0} \\ \nabla \times \mathbf{B} &= \mu_0 \epsilon_0 \frac{\partial \mathbf{E}}{\partial t} + \mu_0 \mathbf{j} \\ \nabla \cdot \mathbf{B} &= 0, \end{aligned}$$

where μ_0 and ϵ_0 are the permeability and permittivity of free space, respectively. $\mathbf{j}$ is the current density and ρ is the charge density.

1.3 Applications

In this project, we care about charged particles in electromagnetic fields. Luckily for us, the Earth produces a dipole magnetic field, the effect of said field extends much further than simply applying a force to a compass. In reality, the magnetic field extends into space and shields us from the incoming solar wind. The solar wind deforms the Earth's dipole field, forming a cavity known as the magnetosphere, which extends to around 10 Earth radii (∼65,000km) sun-ward in average conditions [Shue et al., 1997]. In Figure 1.1 it can be seen that the solar wind is deforming the magnetic field lines (denoted as thin black lines), resulting in the magnetosphere which is shaded in pink. The magnetosphere is largely collisionless, meaning that distinct particle populations at different energies can exist simultaneously. Modelling the motion of charged particles has many practical applications. The biggest is that high-energy particles can damage spacecraft [Eastwood et al., 2017]. The charged particles can be accelerated by waves, greatly increasing their energy (we will not look at this acceleration in this project as it is outside of its scope, however, the numerical solvers that are provided later are capable of solving these problems). The high-energy particles provide many issues to spacecraft, for example, the particles can charge the surface and insides of the craft which can lead to strong discharges [Lanzerotti et al., 1998]. This can damage electronic components and even manifest as phantom commands to the spacecraft [Wilkinson, 1994]. The spacecraft can also lose power due to solar cell degradation caused by energetic particles [Lozinski et al., 2019]. Therefore understanding charged particle dynamics in electromagnetic fields is vital for the safety of spacecraft, and is an extremely active research field.

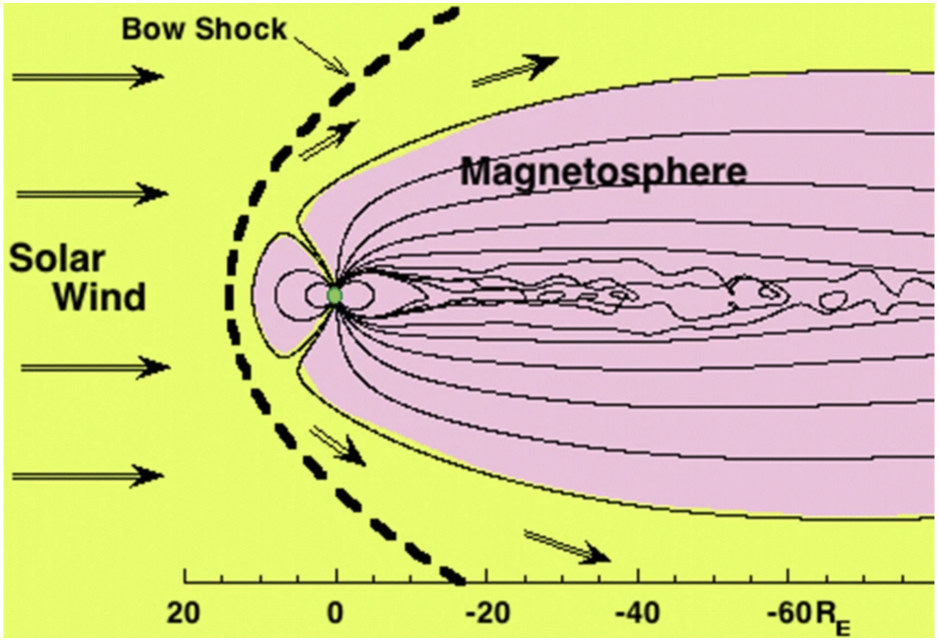


Figure 1.1: A depiction of the Earth's magnetosphere (shaded in pink) and how it is compressed and stretched by the solar-wind plasma (yellow). The solar-wind plasma is flowing from left to right [Borovsky and Valdivia, 2018].

2 Approach

In order to consider the motion of the charged particles the system of coupled ODEs (1.1.3-1.1.5) must be solved. This section will outline the numerical methods of solving used and discuss how the solutions (both analytical and numerical) were implemented computationally. The analytical solutions will be given in Section 3 for their specific EM field.

2.1 Numerical Methods

In this project two self-starting numerical solvers will be used, Euler's Method and the Fourth-order Runge-Kutta method (RK4). Initially, only Euler's Method was included, however, as the complexity of the fields increased an extremely large number of steps, N (or small step size, h) was required to achieve accurate solutions. Hence, the inclusion of the higher order RK4 method.

An overview of both methods and their implementation to the Lorentz Equation will now be provided. Euler's Method is a simple numerical solver that uses the gradient and a sufficiently small step, h, in t to approximate the next value of x. If t_n is some given value of t then the gradient at this point is approximately;

$$\frac{\mathrm{d}x}{\mathrm{d}t} \approx \frac{x(t_n + h) - x(t_n)}{h}.$$

Therefore we can get an estimate of $x(t_n + h)$ by rearranging the equation to

$$x_{n+1} = x_n + hf(x, t), \tag{2.1.1}$$

where x_{n+1} and x_n are the estimates at $t_n + h$ and t_n respectively and $f(x, t)$ is $\frac{\mathrm{d}x}{\mathrm{d}t}$. The initial conditions (x_0, t_0) will be provided. From a physical standpoint, Euler's Method can be conceptualised as tracing the tangent line at each step (each new x_n) to approximate the curve of the solution. This tangent line extends h along the t-axis.

From the Taylor expansion of $x(t_n + h)$ about t_n, we uncover Euler's Method and some error. This is known as the local truncation error (LTE), $T(h)$, and it can be seen to be second-order $O(h^2)$:

$$\begin{aligned} x(t_n + h) &= x(t_n) + hf(x, t) + \frac{h^2}{2} f'(x, t) + O(h^3) \\ \Rightarrow T(h) &= \frac{h^2}{2} f'(x, t) + O(h^3). \end{aligned}$$

Despite a second-order LTE, the global error is still first-order. This is because N steps, of size h, must be taken across the interval. The global error is the accumulation and propagation of local truncation errors at each step. Hence its error will be proportional to the LTE multiplied by N (ie $\frac{1}{h}$). This means the global error is first-order, $O(h)$, as the error will accumulate from each step and the largest error will occur at the end of the interval (see Figure 2.1).

In implementing Euler's Method for the Lorentz equation, we use the fundamental relationship between position ($\mathbf{r}$), velocity ($\mathbf{v}$), and acceleration ($\mathbf{a}$). Specifically, we recognise that the derivative of position with respect to time is velocity, and the derivative of velocity with respect to time is acceleration. Therefore, updating the components of velocity involves utilising the acceleration from the previous step, while updating the components of position involves utilising the velocity from the previous step. This is seen below for the $\hat{\mathbf{x}}$-component of $\mathbf{r}$ and $\mathbf{v}$;

$$x_{n+1} = x_n + hv_{x_n} \tag{2.1.2}$$

$$v_{x_{n+1}} = v_{x_n} + ha_{x_n} \tag{2.1.3}$$

$$t_{n+1} = t_n + h. \tag{2.1.4}$$

It should be noted that the force from the Lorentz Equation will need to be updated for each time step. This is because the velocity of the particle is changing due to the force affecting its direction. From examining Equation (1.1.2), a change in velocity will affect the acceleration. Having the acceleration update at each time step is also vital when dealing with inhomogeneous or time-varying fields, as the applied fields will change with position or time, meaning the acceleration from the Lorentz Force will be dependent on the position of the particle or the time elapsed. This method was implemented computationally using Python as follows:

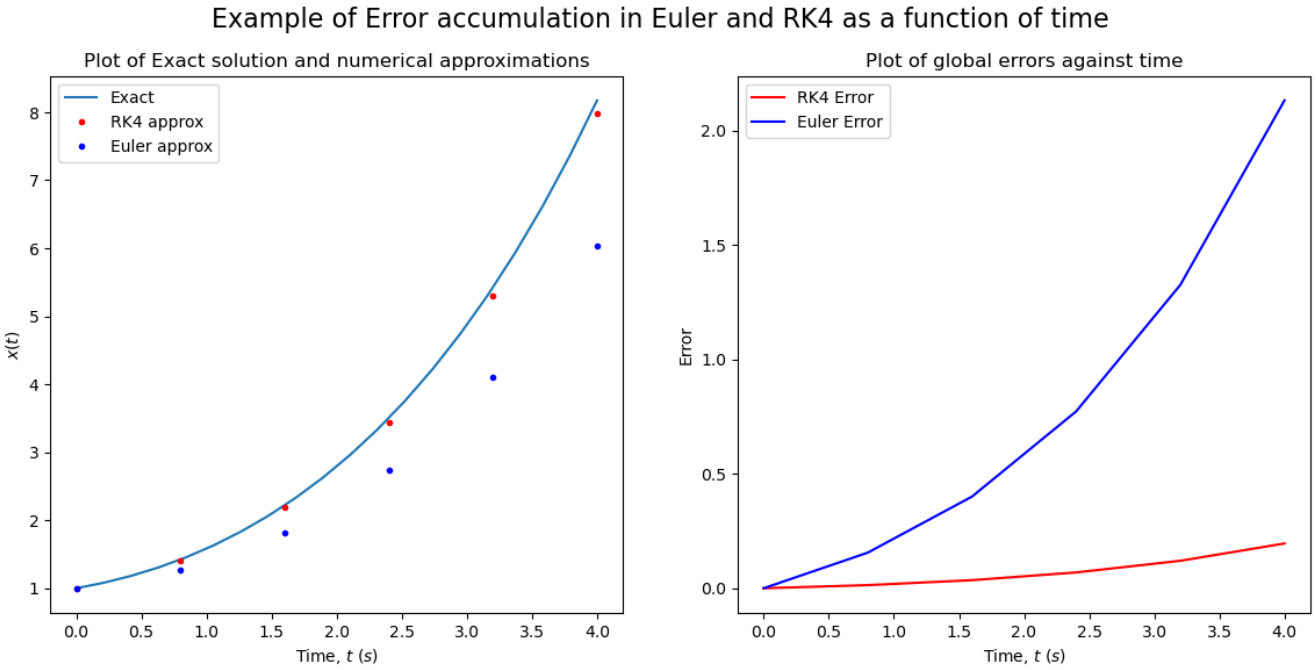


Figure 2.1: Graphs of numerical solvers on a given exponential function. The time step was chosen to be deliberately large to exacerbate the error.

```
# Define function that will give the Acceleration due to Lorentz Force
def lorentz_acc(x, y, z, v_x, v_y, v_z, t):
        # Calculate the components of the Lorentz force (electric + magnetic)
        ax = q/m * (Ex(x,y,z,t) + ((v_y * Bz(x,y,z,t)) - (v_z * By(x,y,z,t))))
        ay = q/m * (Ey(x,y,z,t) + ((v_z * Bx(x,y,z,t)) - (v_x * Bz(x,y,z,t))))
        az = q/m * (Ez(x,y,z,t) + ((v_x * By(x,y,z,t)) - (v_y * Bx(x,y,z,t))))
        return ax, ay, az

def euler_approx(N,x_0,y_0,z_0,v_x,v_y,v_z,B_x,B_y,B_z,E_x,E_y,E_z,q,m,t_end):
    # Calculate step size
    h = (t_end)/(N-1)

```

```
    # Assigning the first value of each component in x,y, and z to match IC
    x = x_0
    y = y_0
    z = z_0

    # Stores for position of particle
    xvals = [x]
    yvals = [y]
    zvals = [z]

    t = 0

    for i in range(N-1):
        # Components of acceleration at time t for the current position
        a_x,a_y,a_z = lorentz_acc(x,y,z,v_x,v_y,v_z,t)
        # Update position using its derivative (velocity)
        x += (v_x * h)
        y += (v_y * h)
        z += (v_z * h)
        # Update velocity using its derivative (acceleration)
        v_x += (a_x * h)
        v_y += (a_y * h)
        v_z += (a_z * h)
        xvals.append(x) # appending a list of x for plot
        yvals.append(y)
        zvals.append(z)
        # Update time by h
        t += h

    # Convert lists to numpy arrays for convenience
    xvals = np.array(xvals)
    yvals = np.array(yvals)
    zvals = np.array(zvals)
    return xvals, yvals, zvals
```

The fourth-order Runge-Kutta method is a modification of Euler's method but follows the same basic principles. In fact, Euler's method is the first-order Runge-Kutta method. To improve upon Euler's method, RK4 takes a weighted average of the gradient between $x(t_n)$ and $x(t_n + h)$. This means the accuracy is greatly improved in comparison to Euler's Method where a single measure of gradient from the tangent at x_n is extended by the step size. The intermediate steps taken to achieve the weighted average of the slope are as follows:

$$\begin{aligned} K_1 &= hf(x_n, t_n) \\ K_2 &= hf(x_n + \frac{1}{2}K_1, t_n + \frac{1}{2}h) \\ K_3 &= hf(x_n + \frac{1}{2}K_2, t_n + \frac{1}{2}h) \\ K_4 &= hf(x_n + K_3, t_n + h), \end{aligned}$$

then the estimate for x_{n+1} is:

$$x_{n+1} = x_n + \frac{1}{6}\left(K_1 + 2K_2 + 2K_3 + K_4\right) \tag{2.1.5}$$

$$t_{n+1} = t_n + h. \tag{2.1.6}$$

As its name suggests the global truncation error scales with the fourth power of step size, $O(h^4)$. As anticipated the local truncation error will be one order higher at $O(h^5)$. The RK4 method assumes that the function it is approximating has, at least, continuous fourth-order derivatives. If the function does not meet this smoothness criterion, the method may not achieve fourth-order accuracy. However, the analytical solutions that this method will be approximating in this project include trigonometric functions, thus this criterion is satisfied.

The RK4 method is implemented in a similar way to Euler's method - using the time derivatives of position and velocity. However, great care must be taken as the acceleration from the Lorentz Force must be updated with each intermediate step, ie before calculating K_1, K_2, K_3 and K_4. In practise, for the x component of position and velocity, this looks like:

$$\begin{aligned}
K_{1_x} &= hv_x(x_n, t_n) \\
K_{2_x} &= hv_x(x_n + \frac{1}{2}K_{1_x}, t_n + \frac{1}{2}h) \\
K_{3_x} &= hv_x(x_n + \frac{1}{2}K_{2_x}, t_n + \frac{1}{2}h) \\
K_{4_x} &= hv_x(x_n + K_{3_x}, t_n + h) \\
K_{1_{v_x}} &= ha_x(x_n, t_n) \\
K_{2_{v_x}} &= ha_x(x_n + \frac{1}{2}K_{1_{v_x}}, t_n + \frac{1}{2}h) \\
K_{3_{v_x}} &= ha_x(x_n + \frac{1}{2}K_{2_{v_x}}, t_n + \frac{1}{2}h) \\
K_{4_{v_x}} &= ha_x(x_n + K_{3_{v_x}}, t_n + h),
\end{aligned}$$

therefore, the estimates for x_{n+1} and $v_{x_{n+1}}$ are:

$$x_{n+1} = x_n + \frac{1}{6}\left(K_{1_x} + 2K_{2_x} + 2K_{3_x} + K_{4_x}\right) \tag{2.1.7}$$

$$v_{x_{n+1}} = v_{x_n} + \frac{1}{6}\left(K_{1_{v_x}} + 2K_{2_{v_x}} + 2K_{3_{v_x}} + K_{4_{v_x}}\right) \tag{2.1.8}$$

$$t_{n+1} = t_n + h. \tag{2.1.9}$$

This method was implemented computationally using Python as follows:

```
def rk4_approx(N, x_0, y_0, z_0, v_x, v_y, v_z, B_x, B_y, B_z, E_x, E_y, E_z, q, m, t_end):
    # Calculate step size
    h = t_end / (N - 1)

    # Assigning the first value of each component in x,y, and z to match IC
    x = x_0
    y = y_0
    z = z_0

    # Stores for position of particle
    xvals = [x]
    yvals = [y]
    zvals = [z]

    t = 0

    for i in range(N-1):
        # Current accelerations
        a_x, a_y, a_z = lorentz_acc(x,y,z,v_x,v_y,v_z,t)

```

```
        # K1
        k1_vx, k1_vy, k1_vz = h*a_x, h*a_y, h*a_z
        k1_x, k1_y, k1_z = h*v_x, h*v_y, h*v_z

        # K2
        # Update the acceleration from Lorentz Force
        ax_mid1, ay_mid1, az_mid1 = lorentz_acc(    # Linebreak due to long function input
            x+(0.5*k1_x),y+(0.5*k1_y),z+(0.5*k1_z),
            v_x+(0.5*k1_vx),v_y+(0.5*k1_vy),v_z+(0.5*k1_vz),t+0.5*h)

        k2_vx, k2_vy, k2_vz = h*ax_mid1, h*ay_mid1, h*az_mid1
        k2_x, k2_y, k2_z = h*(v_x + 0.5*k1_vx), h*(v_y + 0.5*k1_vy), h*(v_z + 0.5*k1_vz)

        # K3
        # Update the acceleration from Lorentz Force
        ax_mid2, ay_mid2, az_mid2 = lorentz_acc(
            x+(0.5*k2_x),y+(0.5*k2_y),z+(0.5*k2_z),
            v_x+(0.5*k2_vx),v_y+(0.5*k2_vy),v_z+(0.5*k2_vz),t+0.5*h)

        k3_vx, k3_vy, k3_vz = h * ax_mid2, h * ay_mid2, h * az_mid2
        k3_x, k3_y, k3_z = h * (v_x + 0.5*k2_vx), h * (v_y + 0.5*k2_vy), h * (v_z + 0.5*k2_vz)

        # K4
        # Update the acceleration from Lorentz Force
        ax_end, ay_end, az_end = lorentz_acc(
            x+(k3_x),y+(k3_y),z+(k3_z),v_x+(k3_vx),v_y+(k3_vy),v_z+(k3_vz),t+h)

        k4_vx, k4_vy, k4_vz = h * ax_end, h * ay_end, h * az_end
        k4_x, k4_y, k4_z = h * (v_x + k3_vx), h * (v_y + k3_vy), h * (v_z + k3_vz)

        # Update velocity and position
        v_x += (k1_vx + (2*k2_vx) + (2*k3_vx) + k4_vx) / 6
        v_y += (k1_vy + (2*k2_vy) + (2*k3_vy) + k4_vy) / 6
        v_z += (k1_vz + (2*k2_vz) + (2*k3_vz) + k4_vz) / 6

        x += (k1_x + (2*k2_x) + (2*k3_x) + k4_x) / 6
        y += (k1_y + (2*k2_y) + (2*k3_y) + k4_y) / 6
        z += (k1_z + (2*k2_z) + (2*k3_z) + k4_z) / 6

        # Append the new values to the lists
        xvals.append(x)
        yvals.append(y)
        zvals.append(z)

        t += h

    # Convert lists to numpy arrays for convenience
    xvals = np.array(xvals)
    yvals = np.array(yvals)
    zvals = np.array(zvals)

    return xvals, yvals, zvals
```

As both the Euler and RK4 solvers I programmed update the applied fields (and thus the resultant acceleration from the Lorentz Equation) at every time step, they can be used to solve the particle trajectory in the most

complex EM fields that vary both in time and position. All of the code used for modelling can be found in the appendix.

2.2 Non-dimensionalisation

When dealing with the physics of charged particles the magnitudes of terms are oftentimes extremely small (below machine precision) or extremely large. This creates a problem as a computer is not able to handle these numbers. To combat this, all terms that appear within the Lorentz Equation will be normalised. Therefore, the values inputted into Python will be dimensionless but will represent some multiple of a normalised value that will be stated later. It is important to note that these normalisation factors cannot be chosen at will, the quantities are linked by their SI units and thus the values of normalisation factors are dependant on other terms. For example, if a normalised length was set to be $L_0 = 5\text{m}$ and time as $T_0 = 2\text{s}$, then the normalisation factor for velocity is predetermined by these previous choices as $v_0 = \frac{L_0}{T_0}$, hence $v_0 = \frac{5}{2}\text{ms}^{-1}$. The values chosen for normalisation are somewhat related to the real-world applications of the Lorentz Equation in Earth's magnetosphere. This means the particle trajectories will be accurate in terms of Earth's Magnetosphere.

The following terms will be normalised, dimensionless quantities are represented by $\tilde{}$.

$$\begin{aligned}
L &= \tilde{L}L_0, \text{ Length, SI Unit: [m]} \\
T &= \tilde{T}T_0, \text{ Time, SI Unit: [s]} \\
v &= \tilde{v}v_0, \text{ Velocity, SI Units: } \left[\text{m}\cdot\text{s}^{-1}\right] \\
q &= \tilde{q}q_0, \text{ Charge, SI Units: [A}\cdot\text{s]} \\
m &= \tilde{m}m_0, \text{ Mass, SI Unit: [kg]} \\
B &= \tilde{B}B_0, \text{ Magnetic field, SI Units: } \left[\text{kg}\cdot\text{A}^{-1}\cdot\text{s}^{-2}\right] \\
E &= \tilde{E}E_0, \text{ Electric field, SI Units: } \left[\text{kg}\cdot\text{m}\cdot\text{A}^{-1}\cdot\text{s}^{-3}\right] \\
W &= \tilde{W}W_0, \text{ Kinetic Energy, SI Units: } \left[\text{kg}\cdot\text{m}^2\cdot\text{s}^{-2}\right],
\end{aligned}$$

where m is metres, s is seconds, A is amperes and kg is kilograms.
Firstly, for simplicity, the elementary charge and the mass of an electron will be used as normalisation factors, such that

$$\begin{aligned}
q_0 &= 1.60217663 \times 10^{-19}\ \text{C} \\
m_0 = m_e &= 9.109383702 \times 10^{-31}\ \text{kg}
\end{aligned}$$

To ensure a similar magnetic field strength to Earth's magnetosphere we will use real-world values of it. The magnetic field at the magnetic equator from Degeling et al. [2007] will be used, $B_{eq} = 3.11 \times 10^{-5}\text{T}$. The field will vary according to

$$B(r) = B_{eq}\left(\frac{R_E}{r}\right)^3,$$

and we chose to normalise at $r = 5R_E$, where $R_E = 1$ Earth radii $= 6371\text{km}$. The magnetic field strength is:

$$\begin{aligned}
B_0 = B(5) &= 3.11 \times 10^{-5}\left(\frac{1}{5}\right)^3\ \text{T} \\
&= 2.488 \times 10^{-7}\ \text{T}
\end{aligned}$$

We will also normalise the kinetic energy of the particle, for simplicity, by one electron volt, hence

$$W_0 = 1\text{eV} = 1.60217663 \times 10^{-19}\ \text{J}$$

From these chosen normalisation quantities the remaining ones can be deduced via relation from SI units. Using the SI units of magnetic field, the normalised time, T_0, can be calculated. Note that units of current depend on quantities already normalised:

$$\begin{aligned} B_0 &= \frac{m_0}{A_0 {T_0}^2} = \frac{m_0}{\left(\frac{q_0}{T_0}\right) {T_0}^2} \\ \Rightarrow T_0 &= \frac{m_0}{q_0 B_0} \\ &= \frac{9.109383702 \times 10^{-31}}{1.60217663 \times 10^{-19} \cdot 2.488 \times 10^{-7}} \text{ s} \\ &= 2.285 \times 10^{-5} \text{ s} \end{aligned}$$

It is now possible to determine the normalising velocity, v_0, from the dimensions of energy

$$\begin{aligned} W_0 &= m_0 \left(\frac{L_0}{T_0}\right)^2 = m_0 {v_0}^2 \\ \Rightarrow v_0 &= \sqrt{\frac{W_0}{m_0}} \\ &= \sqrt{\frac{1.60217663 \times 10^{-19}}{9.109383702 \times 10^{-31}}} \text{ ms}^{-1} \\ &= 4.194 \times 10^5 \text{ ms}^{-1} \end{aligned}$$

The normalised length can now be determined via the normalised time and velocity

$$\begin{aligned} L_0 &= v_0 T_0 \\ &= 4.194 \times 10^5 \cdot 2.285 \times 10^{-5} \text{ m} \\ &= 9.583 \text{ m} \end{aligned}$$

Finally, the normalised electric field strength can be determined via its dimensions in SI units. This is because we have already normalised all the terms that are in its SI units

$$\begin{aligned} E_0 &= \frac{m_0 L_0}{A_0 \cdot {T_0}^3} = \frac{m_0 L_0}{q_0 \cdot {T_0}^2} \\ &= \frac{9.109383702 \times 10^{-31} \cdot 9.583}{1.60217663 \times 10^{-19} \cdot (2.285 \times 10^{-5})^2} \text{ NC}^{-1} \\ &= 0.104 \text{ NC}^{-1} \end{aligned}$$

2.3 Computational Application

Implementing the numerical solver code above is straightforward. The functions `euler_approx` and `rk4_approx` both output arrays of position for given dimensionless initial conditions. The length of the arrays will be specified by the number of steps the function should take. Analytical solutions, once solved, can be represented by functions in Python, for example:

```
1 def z_exact(z0, vz, t):
2     z = vz * t + z0
3     return z
```

Therefore, when the relevant initial conditions and an array of time values are given, an array of the particle's position in the specified component is returned.

2.4 Error

An integral part of this project is the comparison between numerical and analytical solutions. As discussed earlier in this section, the error between the two solution types will vary with the step size, h, within a given interval (it will also vary with the number of steps, N, as these are proportional to the inverse of one another). Therefore, to measure the error, an array of different values of N will be looped through. For each value of N, the maximum error in position between the numerical and analytical solution will be taken. The maximum error will occur at the latest time (ie the end position). This is because the local truncation error from each step will accumulate throughout the method, resulting in the final value having the largest error. This is helpful as it means the final position values for the analytical and numerical solutions can be compared and the distance between them is the global error of the numerical method. The error is calculated for each value of N. Plots of error as a function of h and N are then produced. The implementation of this can be seen below:

```
Nvals = np.arange(1000,20000,1000)
Err_vals_e = []
Err_vals_rk = []
hvals = []

for N in Nvals:
    tvals = np.linspace(0,t_max,N)

    h = t_max/(N-1)
    hvals.append(h)

    x1,y1,z1=solver(x_exact,y_exact,z_exact,N,x_0,y_0,z_0,v_x,v_y,v_z,B_x,B_y,B_z,E_x,E_y,E_z,q,m,t_max)
    x2,y2,z3=euler_approx(N,x_0,y_0,z_0,v_x,v_y,v_z,B_x,B_y,B_z,E_x,E_y,E_z,q,m,t_max)
    x3,y3,z3=rk4_approx(N,x_0,y_0,z_0,v_x,v_y,v_z,B_x,B_y,B_z,E_x,E_y,E_z,q,m,t_max)

    # Maximum error will be the final value as Global
    err_euler = np.sqrt((x2[-1]-x1[-1])**2 + (y2[-1]-y1[-1])**2 + (z2[-1]-z1[-1])**2)
    err_rk4 = np.sqrt((x3[-1]-x1[-1])**2 + (y3[-1]-y1[-1])**2 + (z3[-1]-z1[-1])**2)

    Err_vals_e.append(err_euler)
    Err_vals_rk.append(err_rk4)
```

Where `x_exact`, `y_exact`, and `z_exact` are the functions that define the analytical solution in position for the given component.

As the error accumulates, the error in position will increase with each time step that is taken. It can be insightful to look at the error in each component of the particle's position as a function of time. This was done by setting a value for the number of steps in the interval, N, and plotting the absolute value of the difference between numerical and analytical position for each compenent. The computational implementation can be seen below:

```
N = 5000 # Sufficent N so both schemes are accurate
tvals = np.linspace(0,t_max,N)

# Use solvers to obtain arrays of position
x1,y1,z1 = solver(x_exact1,y_exact1,z_exact,N,x_0,y_0,z_0,v_x,v_y,v_z,B_x,B_y,B_z,E_x,E_y,E_z,q,m,t_max)
x2,y2,z2 = euler_approx(N,x_0,y_0,z_0,v_x,v_y,v_z,B_x,B_y,B_z,E_x,E_y,E_z,q,m,t_max)
x3,y3,z3 = rk4_approx(N,x_0,y_0,z_0,v_x,v_y,v_z,B_x,B_y,B_z,E_x,E_y,E_z,q,m,t_max)

# Find error in position for Euler and RK4
error_euler = np.sqrt((x2-x1)**2 + (y2-y1)**2 + (z2-z1)**2)
error_rk4 = np.sqrt((x3-x1)**2 + (y3-y1)**2 + (z3-z1)**2)
```

```
12
13  # Find error in x, y, and z for Euler and RK4
14  error_xe = abs(x2-x1)
15  error_ye = abs(y2-y1)
16  error_ze = abs(z2-z1)
17  error_xrk = abs(x3-x1)
18  error_yrk = abs(y3-y1)
19  error_zrk = abs(z3-z1)
```

3 Particle Motion in Electromagnetic Fields

This section will focus on analytically solving the coupled second-order ODEs outlined in equations (1.1.3)-(1.1.5). The behaviour of the solution will be examined for various initial conditions and then it will be compared to the numerical solution, undergoing error analysis.

3.1 Constant Magnetic Field

We will begin by considering one of the simplest applied fields, a spatially uniform magnetic field, $\mathbf{B}$, acting in the $\hat{\mathbf{z}}$-direction with $\mathbf{E} = \mathbf{0}$. Let $\mathbf{B} = (0, 0, B_z)$ and substitute these field values into equations (1.1.3)-(1.1.5), resulting in the following system of ODEs:

$$\ddot{x} = \frac{q}{m}(\dot{y}B_z) = \Omega\dot{y} \tag{3.1.1}$$

$$\ddot{y} = \frac{q}{m}(-\dot{x}B_z) = -\Omega\dot{x} \tag{3.1.2}$$

$$\ddot{z} = \frac{q}{m}(0) = 0, \tag{3.1.3}$$

where $\Omega = \frac{qB_z}{m}$ is the gyro-frequency of particle orbit. This is the frequency with which a particle gyrates about a magnetic field line. It can be seen that Ω is indeed a frequency by analysing the SI units of q, B_z, and m and seeing that the expression for gyro-frequency reduces to units of s^{-1} as required! In order to solve equations (3.1.1) and (3.1.2) the kinetic energy of the particle must be considered. By substituting $\mathbf{E} = 0$ into the Lorentz Equation (1.1.1), it becomes:

$$m\ddot{\mathbf{r}} = q(\dot{\mathbf{r}} \times \mathbf{B}). \tag{3.1.4}$$

Hence, by properties of the cross product, velocity will be perpendicular to acceleration, therefore:

$$m\ddot{\mathbf{r}} \cdot \dot{\mathbf{r}} = 0, \tag{3.1.5}$$

integrating equation (3.1.5) implies that,

$$\frac{1}{2}m\dot{r}^2 = const. = W. \tag{3.1.6}$$

This can be understood by considering the work done on the particle. As the force from the Lorentz Equation is always perpendicular to the velocity of the particle in this field, no work is done, thus W must be constant. This is true for any $\mathbf{B}(\mathbf{r})$ when $\mathbf{E} = \mathbf{0}$. It is possible to break the total kinetic energy down into components parallel ($W_\parallel$) and perpendicular ($W_\perp$) to the magnetic field. Hence,

$$W = W_\parallel + W_\perp = \frac{1}{2}m\left(v_\parallel^2 + v_\perp^2\right) = const. \tag{3.1.7}$$

This implies that $v_\perp$ is constant in this field. We will need to apply this condition to find the oscillatory velocity of the particle's motion.

We will solve equations (3.1.1) and (3.1.2) with the initial velocity being $\mathbf{v} = (v_x, 0, v_z)$, therefore $v_\perp = v_x$. The particle will start at $\mathbf{r} = (x_0, y_0 - r_L, z_0)$.To uncouple the system of equations we will differentiate equations (3.1.1) and (3.1.2) with respect to time:

$$\dddot{x} = \Omega\ddot{y} \tag{3.1.8}$$

$$\dddot{y} = -\Omega\ddot{x}, \tag{3.1.9}$$

substituting equations (3.1.1) and (3.1.2) into the above results in two second-order uncoupled ODEs for $\dot{x}$ and $\dot{y}$

$$\ddot{\dot{x}} + \Omega^2 \dot{x} = 0 \tag{3.1.10}$$

$$\ddot{\dot{y}} + \Omega^2 \dot{y} = 0. \tag{3.1.11}$$

Which can be solved for $\dot{x}$ and $\dot{y}$:

$$\dot{x} = v_\perp \cos(\Omega t) \tag{3.1.12}$$

$$\dot{y} = -v_\perp \sin(\Omega t). \tag{3.1.13}$$

Note the oscillatory velocity, $v_\perp$, is a constant. This is due to the conservation of $W_\perp$, with these initial conditions $v_\perp = v_x$. Therefore, the magnitude of the velocity perpendicular to the magnetic field is constant as required. Integrating equations (3.1.12) and (3.1.13) with respect to time will provide us with the x and y position of the charged particle,

$$x = \frac{v_\perp}{\Omega} \sin(\Omega t) + x_0 \tag{3.1.14}$$

$$y = \frac{v_\perp}{\Omega} \cos(\Omega t) + y_0. \tag{3.1.15}$$

The solution to (3.1.3) is trivial and is given by:

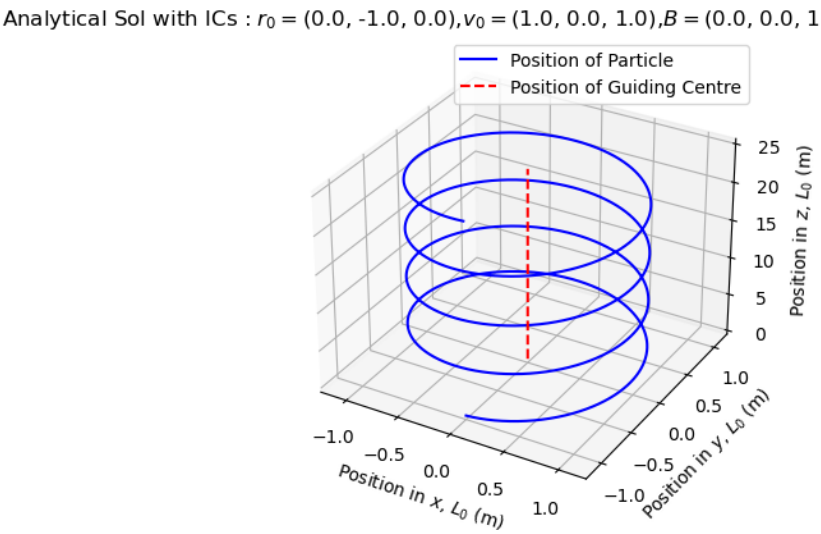
$$z = v_\parallel t + z_0, \tag{3.1.16}$$

where $v_\parallel = v_z = const$, is the component of velocity parallel to the magnetic field.
This is the full analytical solution for a charged particle in a homogeneous magnetic field acting in the $\hat{\mathbf{z}}$-direction with no electric field. It describes a helix of constant pitch, centred on (x_0, y_0) with radius $\frac{v_\perp}{|\Omega|} = r_L$. The radius of the helix, r_L, is often referred to as the Larmor radius. See Figure 3.1 for an illustration of the particle's trajectory. As the gyro-frequency is proportional the the strength of the magnetic field, a particle in a stronger magnetic field will orbit at a faster rate. Moreover, it will have a shorter Larmor radius due to this increased gyro-frequency. This can be seen when comparing Figure 3.1a to Figure 3.1b, where the magnetic field is stronger. When dealing with particle orbits it is useful to consider the motion of the orbits guiding centre. The guiding centre is the average position of the particle throughout its orbit. Therefore in the case of a circular orbit, the guiding centre in the xy-plane will be situated on the centre of the circle and move upwards in the $\hat{\mathbf{z}}$-direction with velocity $v_\parallel$. The guiding centre is denoted as the red dotted line in Figure 3.1. From the analytical solutions and the centre of the orbit, we can describe the position and velocity of the particle's guiding centre as:

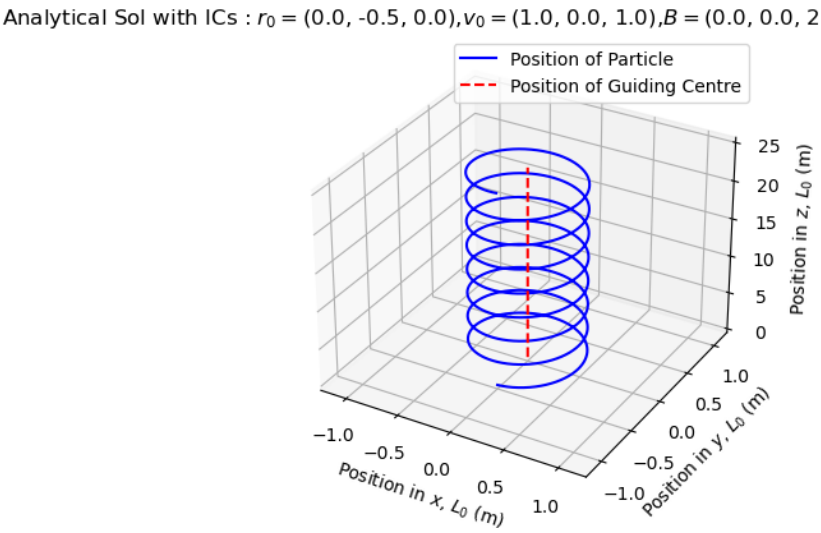
$$\mathbf{r}_g = (x_0, y_0, v_\parallel t + z_0)$$

$$\mathbf{v}_g = (0, 0, v_\parallel).$$

We designate y_0 as the initial position of the guiding centre rather than the particle's starting point. This distinction is crucial, especially when dealing with more complex fields where Taylor expansions are necessary around the guiding centre. Therefore it is good practice to denote y_0 as such throughout. This explains why the initial position of the particle was denoted as $y_0 - r_L$.



(a) Orbit of an electron in a magnetic field similar to that found in the magnetosphere.



(b) Orbit of an electron in a stronger magnetic field, hence increasing its gyro-frequency.

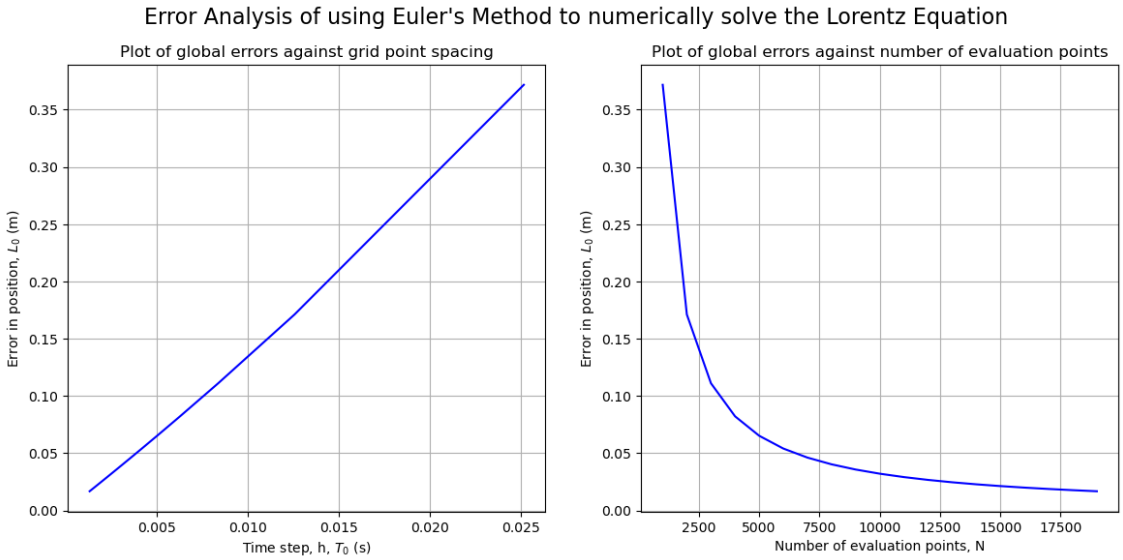
Figure 3.1: Particle trajectories of an electron, with identical initial velocities, in differing magnetic fields.

To account for a particle's phase within its orbit, we can introduce a phase shift (α). This adjustment allows us to describe the initial velocity perpendicular to the magnetic field, $v_\perp$, as a combination of both v_x and v_y, effectively situating the particle at different starting points within its orbital cycle. To incorporate the phase of the particle into our analytical solution Equations (3.1.14) and (3.1.15) become:

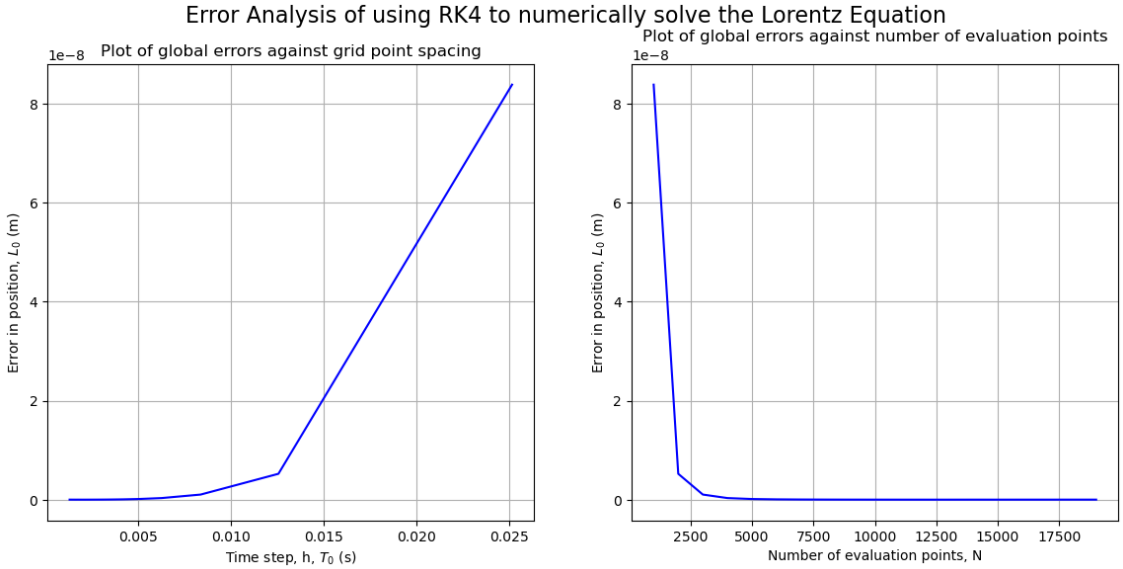
$$\begin{aligned} x &= \frac{v_\perp}{\Omega}\sin(\Omega t + \alpha) + x_0 \\ y &= \frac{v_\perp}{\Omega}\cos(\Omega t + \alpha) + y_0 \end{aligned}$$

For example with the current initial conditions the charged particle will start at the bottom of the circle representing its orbit, ie $v_y = 0$. With $\alpha \neq 0$ then $v_\perp = \sqrt{v_x^2 + v_y^2}$ and the particle will start at a different point in its orbit. For the remainder of this project we not will include the phase shift for simplicity. This omission is made under the assumption that it is always feasible to align $v_\perp$ with one of the axes perpendicular to the applied magnetic field (for this project's sake that will be the x-axis), thus simplifying our analysis without significantly limiting our understanding of the particle's motion.

As outlined in Section 2 we will now compare the analytical solution derived above to the numerical solvers utilising the Euler Method and Fourth-order Runge-Kutta method respectively. We will first compare the error in the final position of the particle as a function of both the number of time steps, N, and the size of the time steps, h. How error evolves as a function of these variables can be seen in Figure 3.2.



(a) Error of Euler's Method as a function of h and N.



(b) Error of the fourth-order Runge-Kutta method as a function of h and N.

Figure 3.2: Error of numerical solvers for values of h from $\sim$0.0025 to $\sim$0.001 and values of N from 1000 to 20000 for a particle and fields with initial conditions consistent with that seen in Figure 3.1a.

These graphs confirm that the global error the solvers produce is consistent with the relationships outlined in Section 2.1. In other words, the global error in Euler's method is proportional to step size, h (hence the straight line), and inversely related to the number of steps taken, N. In comparison, the higher order RK4 method has a smaller global error that scales with h^4 or $\frac{1}{N^4}$. In general, neither method was particularly sensitive to the initial conditions selected. As previously discussed, if the particle was allowed to orbit for a longer time then the error would increase, if step size stayed constant. If the field were to become extremely strong, increasing the gyro-frequency, Ω, there is the potential for some stability issues due to limitations both methods have when dealing with highly oscillatory solutions. However, since the normalised magnetic field is set to be consistent with Earth's magnetosphere these highly oscillatory solutions will not occur for our practical applications of the methods.

We will also analyse the global error as a function of time for the RK4 method, seen in Figure 3.3. The oscillatory nature of the error is due to the functions it is numerically approximating. Equations (3.1.14) and (3.1.15) are both sinusoidal functions and oscillate in time, it is a common property of the Runge-Kutta methods that their error also oscillates with the functions. The phase shift between $\cos(\Omega t)$ and $\sin(\Omega t)$ is the reason the errors are out of phase. Note that despite there being a minimum in the error of the y-component's position the overall error in position is still at a maximum as expected, which can be seen in the final graph. Furthermore, the global error of the overall position of the particle does not oscillate and grows in a similar manner to the graphs shown in Figure 2.1.

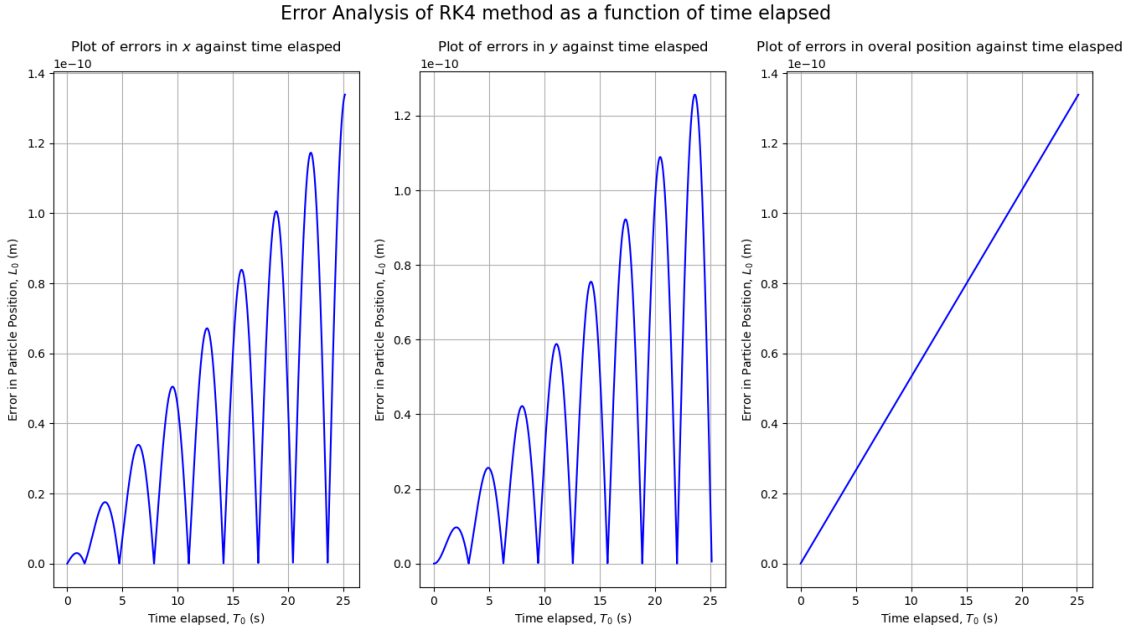


Figure 3.3: Global Error as a function of time for the x-component of position, y-component of position and overall position with $N = 5000$ for a particle and fields with initial conditions consistent with that seen in Figure 3.1a.

3.2 Constant Magnetic and Electric Fields

To add another layer of complexity to the applied EM field, we will now consider a spatially uniform electric field that is perpendicular to the magnetic field applied in Section 3.1. Therefore the applied fields are $\mathbf{B} = (0, 0, B_z)$ and $\mathbf{E} = (0, E_y, 0)$. Therefore, (1.1.3)-(1.1.5) now become:

$$\ddot{x} = \Omega\dot{y} \tag{3.2.1}$$

$$\ddot{y} = \frac{q}{m}E_y - \Omega\dot{x} \tag{3.2.2}$$

$$\ddot{z} = 0. \tag{3.2.3}$$

These equations are solved in the same way as in Section 3.1, with initial conditions of $\mathbf{v} = (v_x, 0, v_\parallel)$ and $\mathbf{r} = (x_0, y_0 - r_L, z_0)$. Once again it is important to consider the kinetic energy of the charged particle. The overall kinetic energy is no longer conserved as $\mathbf{E} \neq \mathbf{0}$, however, the rotational kinetic energy caused by the magnetic field is still conserved for reasons discussed in the previous subsection. Therefore, the magnitude of the particle's oscillatory velocity, u, will not change with time. Similarly, the radius of the particle's orbit will also be constant. However, there will now be a drift caused by the non-zero electric field, meaning the orbits will not be circular.

Differentiating equation (3.2.1) again with respect to time and decoupling leaves us with an inhomogeneous second-order differential equation for $\dot{x}$:

$$\dddot{x} + \Omega^2\dot{x} = \frac{q}{m}\Omega E_y. \tag{3.2.4}$$

We have already solved the complementary function in equation (3.1.12) from Section 3.1, but it will have a rotational velocity now equal to u, which is determined via initial conditions.

$$\dot{x}_{CF} = u\cos(\Omega t). \tag{3.2.5}$$

For the particular integral a solution of the form $\dot{x}_{PI} = A$ will be used, where A is a constant that will be determined. Substituting this into equation(3.2.4) gives:

$$\begin{aligned} \Omega^2 A &= \frac{q}{m}\Omega E_y \\ \Rightarrow A &= \frac{qE_y}{m\Omega} = \frac{E_y}{B_z} = v_E. \end{aligned} \tag{3.2.6}$$

Combining $\dot{x}_{CF}$ and $\dot{x}_{PI}$ gives a full analytical solution for $\dot{x}$:

$$\dot{x} = u\cos(\Omega t) + v_E, \tag{3.2.7}$$

where v_E is the drift velocity due to the electric field. It is possible to determine u via the initial conditions:

$$\begin{gathered} \dot{x} = v_x, \text{ at } t = 0 \\ \Rightarrow u\cos(0) + v_E = v_x \\ \Rightarrow u = v_x - v_E. \end{gathered}$$

We can now substitute the expression for $\dot{x}$ in equation (3.2.7), into the equation for $\ddot{y}$ in equation (3.2.2), and

obtain the y-component of the particle's trajectory:

$$\begin{aligned} \ddot{y} &= \frac{q}{m}E_y - \Omega\left[u\cos\left(\Omega t\right) + v_E\right] \\ &= -u\Omega\cos\left(\Omega t\right) \\ \Rightarrow \dot{y} &= -u\sin\left(\Omega t\right), \end{aligned} \tag{3.2.8}$$

noting that $\frac{q}{m}E_y = \Omega v_E$. Finally, the analytical solution for the position of the charged particle is found by integrating equations (3.2.3), (3.2.7) and (3.2.8) with respect to time and applying the initial conditions:

$$x = \frac{u}{\Omega}\sin\left(\Omega t\right) + v_E t + x_0 \tag{3.2.9}$$

$$y = \frac{u}{\Omega}\cos\left(\Omega t\right) + y_0 \tag{3.2.10}$$

$$z = v_{\parallel} t + z_0 \tag{3.2.11}$$

Where $v_{\parallel}$ is the velocity parallel to the magnetic field and will be constant. The drift velocity, v_E, does not depend on the properties of the charged particle. It solely depends on the sizes of the applied fields ($v_E = \frac{E_y}{B_z}$). The drift term means the particle is now drifting across magnetic field lines rather than simply orbiting one as before. Counter-intuitively the drift velocity is orthogonal to both of the applied fields.

It is possible to derive a general expression for the drift due to a uniform $\mathbf{E}$. This is done by making a transformation into the reference frame moving with the drift velocity, v_E, such that $\mathbf{v} = \mathbf{v}_E + \mathbf{v}_{osc}$, where $\mathbf{v}_{osc}$ is the oscillatory velocity of the particle. Substituting the new expression for $\mathbf{v}$ into the Lorentz Equation (1.1.2), results in:

$$\dot{\mathbf{v}}_{osc} = \frac{q}{m}\left(\mathbf{E} + \mathbf{v}_E \times \mathbf{B}\right) + \frac{q}{m}\mathbf{v}_{osc} \times \mathbf{B}, \tag{3.2.12}$$

now we will consider the drifting frame, such that

$$\mathbf{E} + \mathbf{v}_E \times \mathbf{B} = 0. \tag{3.2.13}$$

This can be solved for $\mathbf{v}_E$ by taking the cross product of both terms with $\mathbf{B}$,

$$\mathbf{E} \times \mathbf{B} = -\left(\mathbf{v}_E \times \mathbf{B}\right) \times \mathbf{B}, \tag{3.2.14}$$

then using the following vector identity

$$\left(\mathbf{A} \times \mathbf{B}\right) \times \mathbf{C} = \mathbf{B}\left(\mathbf{C} \cdot \mathbf{A}\right) - \mathbf{A}\left(\mathbf{C} \cdot \mathbf{B}\right),$$

results in the following equation:

$$\mathbf{E} \times \mathbf{B} = \mathbf{v}_E B^2 - \mathbf{B}\left(\mathbf{v}_E \cdot \mathbf{B}\right). \tag{3.2.15}$$

Noting that the second term on the RHS must vanish as the LHS is perpendicular to $\mathbf{B}$. This implies that the drift must be perpendicular to $\mathbf{B}$, this is in line with what we found in our analytical solution. Now equation (3.2.15) can be solved for $\mathbf{v}_E$, leaving the following general expression for the drift velocity due to an electric field.

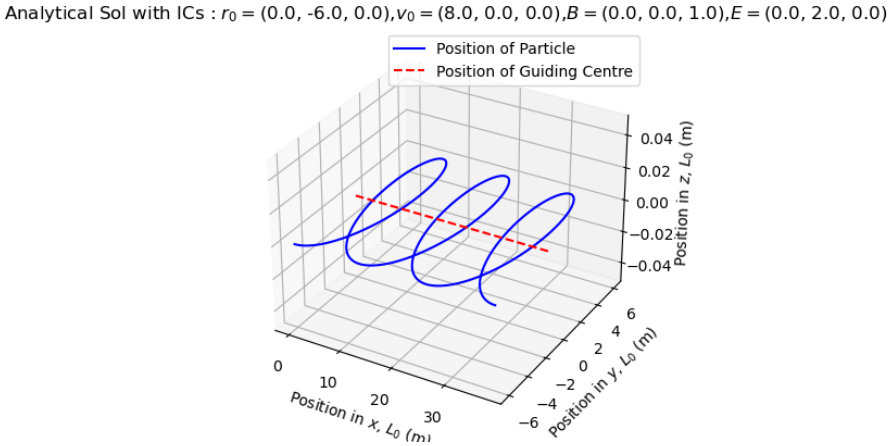
$$\mathbf{v}_E = \frac{\mathbf{E} \times \mathbf{B}}{B^2} \tag{3.2.16}$$

This expression explains the orthogonality between the drift and the applied fields, as the drift is proportional to the cross-product of the fields. Note, in order to keep the non-relativistic approximation true we require $v_E = \frac{E_y}{B_z} \ll c$, this is the case for the fields in Earth's magnetosphere. The velocity and position of the guiding

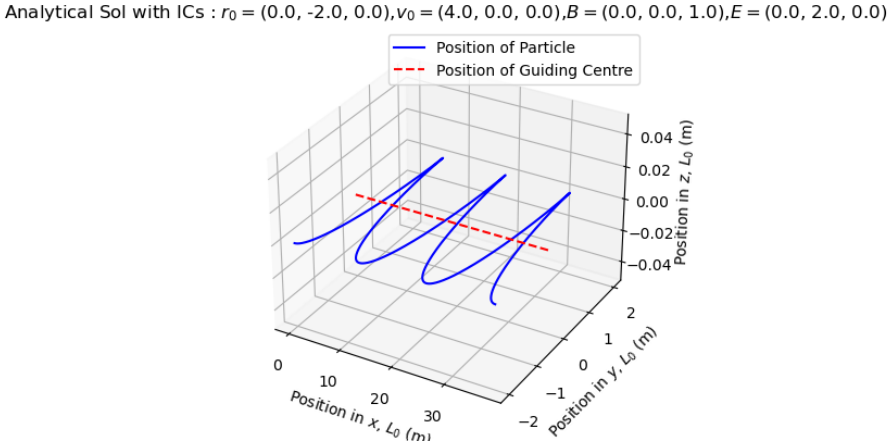
centre, updated to include this drift term are:

$$\begin{aligned} \mathbf{v}_g &= (v_E, 0, v_\parallel) \\ \mathbf{r}_g &= (v_E t + x_0, y_0, v_\parallel t + z_0) \end{aligned}$$

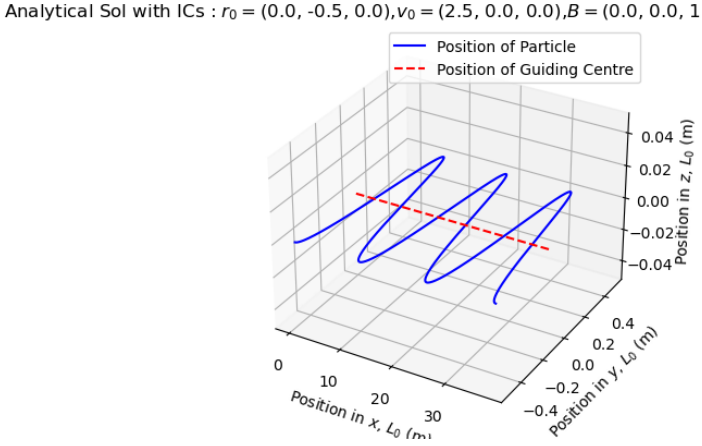
Depending on the value of oscillatory velocity,$u = v_x - v_E$, the orbit of the charged particle can change significantly, this can be seen in Figure 3.4.



(a) Orbit of an electron with oscillatory velocity, u, greater than the drift velocity due to the electric field, v_E.



(b) Orbit of an electron with oscillatory velocity, u, equal to the drift velocity due to the electric field, v_E

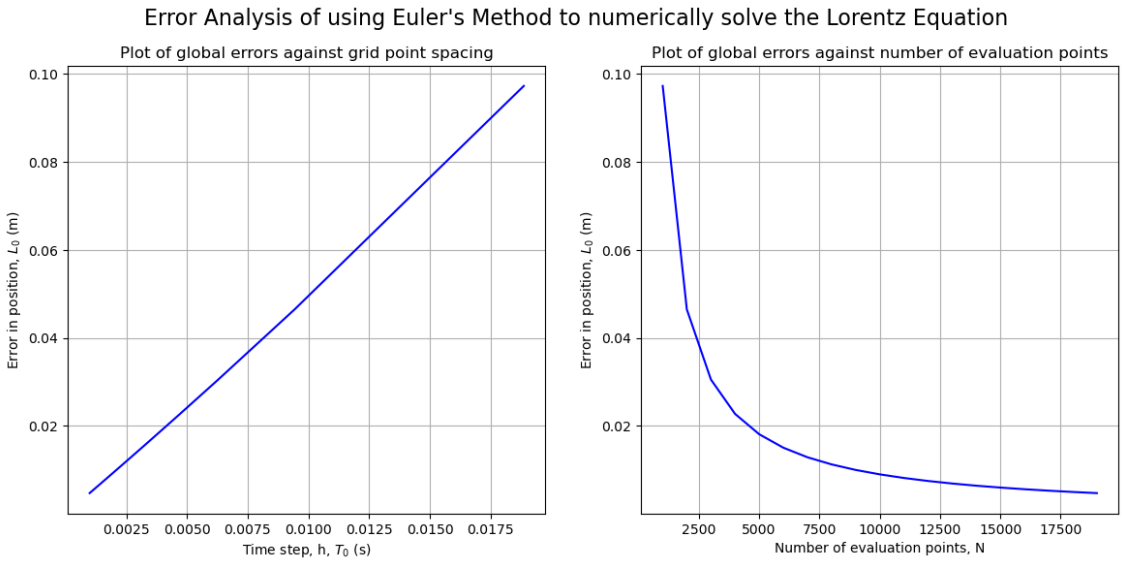


(c) Orbit of an electron with oscillatory velocity, u, less than the drift velocity due to the electric field, v_E

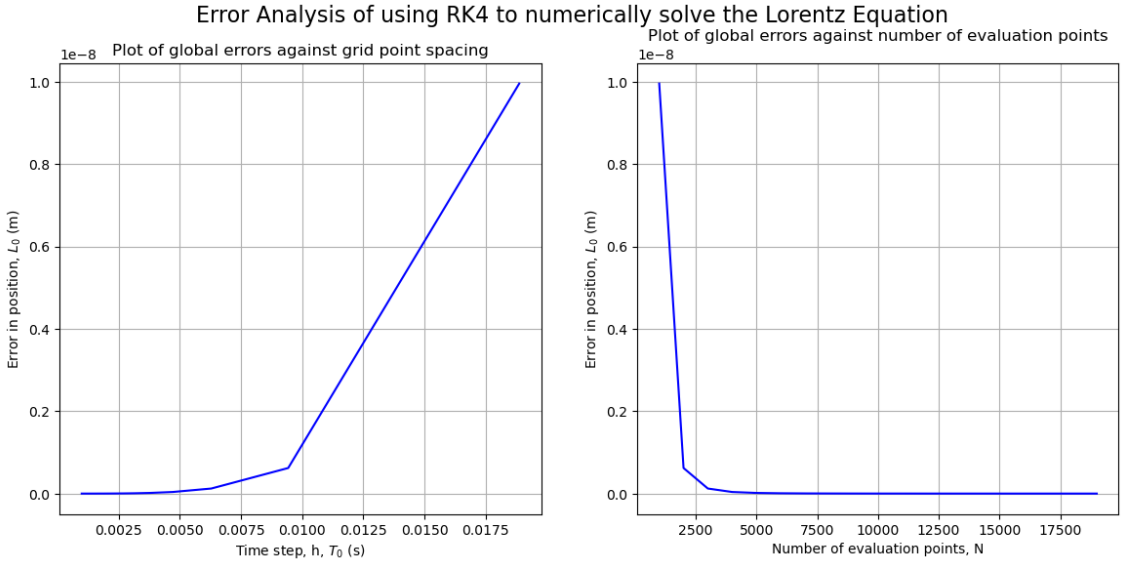
Figure 3.4: Various orbits of an electron, in the same applied fields, depending on the magnitude of their oscillatory velocity, u

The explanation of these different orbits is intuitive when considering the physical perspective of the particle drift. If the oscillatory velocity is greater than the drift velocity, ($u > v_E$), then the particle will complete an orbit of the guiding centre (the thing that is drifting) before it can move the distance of the orbital diameter. This leads to the loops seen in Figure 3.4a. In contrast, when the particle is drifting faster than it is oscillating, ($u < v_E$), then this overlap will not occur and the orbit will resemble a sinusoidal function as seen in Figure 3.4c. If $u = v_E$ there will be no loops as the guiding centre has drifted exactly the orbital diameter, resulting in the discontinuity seen in Figure 3.4b. It can be seen throughout all the subfigures in Figure 3.4 that the drift velocity due to the electric field, v_E, is constant as the magnetic and electric fields are the same for each case.

For the error analysis, the initial condition of the particle will be as given in the $u < v_E$ example. Once again when plotting the global error in position as a function of h and N we obtain the expected convergence rates of the error as seen in Figure 3.5.



(a) Error of Euler's Method as a function of h and N.



(b) Error of the fourth-order Runge-Kutta method as a function of h and N.

Figure 3.5: Error analysis of numerical solvers for values of h from $\sim$0.001 to $\sim$0.019 and values of N from 1000 to 20000 for a particle and fields with initial conditions consistent with that seen in Figure 3.4c.

Similarly as a function of time with a constant step size, the error oscillates. The maximum global error in position is still at the final time step as expected, this can be seen in Figure 3.6. The error for this configuration of electric and magnetic fields also had a low sensitivity to the initial conditions. This, again, is somewhat expected as we are using a very small time step hence the numerical solvers are able to provide accurate solutions despite varying initial conditions.

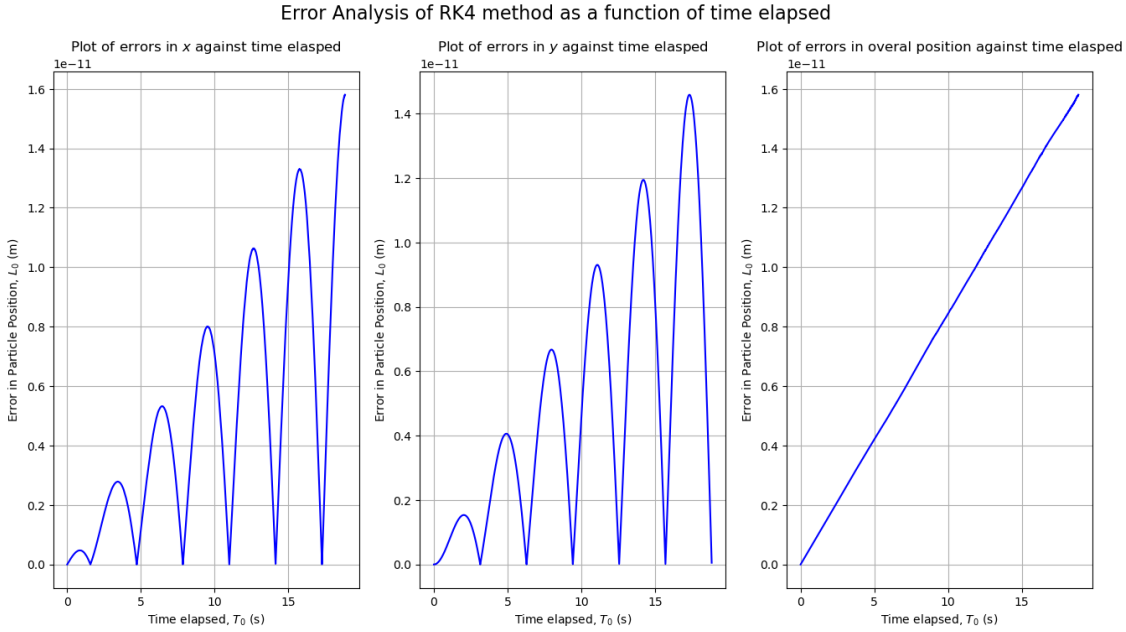


Figure 3.6: Global Error as a function of time for the x-component of position, y-component of position and overall position with $N = 5000$ for a particle and fields with initial conditions consistent with that seen in Figure 3.4c.

3.3 Gradient Drift in an Inhomogeneous Magnetic Field

In reality, it is rare to find a truly constant magnetic field. It will more often than not vary both spatially and with time. When this is the case we are required to solve the Lorentz Equation numerically. However, if the inhomogeneity is small and the field is constant with time it is possible to analytically derive a perturbation solution for the particle trajectory in a spatially varying field. For this we require $\delta\mathbf{B}$, the change in the magnetic field $\mathbf{B}$ over a distance r_L, to be

$$|\delta\mathbf{B}| \ll |\mathbf{B}|,$$

where $|\mathbf{B}|$ is the magnitude of the magnetic field at the guiding centre. In other words, for this approximation, we require that the field the particle experiences during its Larmor orbit is almost constant.

The applied fields in this configuration are $\mathbf{B} = (0, 0, B_z(y))$ and $\mathbf{E} = \mathbf{0}$, therefore, equations (1.1.3)-(1.1.5) now become:

$$\ddot{x} = \Omega(y)\dot{y} \tag{3.3.1}$$

$$\ddot{y} = -\Omega(y)\dot{x} \tag{3.3.2}$$

$$\ddot{z} = 0, \tag{3.3.3}$$

where $\Omega(y) = \frac{q}{m}B_z(y)$. These equations will be solved with the initial conditions of $\mathbf{v} = (v_x, 0, v_\parallel)$ and $\mathbf{r} = (x_0, y_0 - r_L, z_0)$. We will attempt to decouple equations (3.3.1) and (3.3.2) as before by differentiating again

with respect to time. However it must be noted the $\Omega(y)$ is not constant in time and has derivative $\dot{\Omega}(y) = \frac{d\Omega(y)}{dy}\frac{dy}{dt} = \Omega'(y)\dot{y}$, where $\Omega' = \frac{d\Omega(y)}{dy}$. Hence, the equations become:

$$\ddot{x} = \Omega'(y)\dot{y}^2 - \Omega^2(y)\dot{x} \tag{3.3.4}$$

$$\ddot{y} = \Omega'(y)\dot{x}\dot{y} - \Omega^2(y)\dot{y}. \tag{3.3.5}$$

It is possible to perform a Taylor expansion of $\Omega(y)$ about the guiding centre y_0. This is due to the $|\delta\mathbf{B}| \ll |\mathbf{B}|$ assumption as it implies $\Omega(y)$ is almost constant throughout a Larmor orbit. The reason we chose to Taylor expand about the guiding centre is because it provides a stable reference point that more accurately reflects the mean path of the particle's position through the inhomogeneous field in comparison to the particle's instantaneous position or initial position. As the inhomogeneity is small, the particle's orbit will be roughly circular. This means that the guiding centre provides the most accurate approximation of the field the particle will experience in any given position. The Taylor expansion is as follows,

$$\begin{aligned}\Omega(y) &= \Omega(y_0) + (y - y_0)\Omega'(y_0) + O\left(\left(\frac{|\delta\mathbf{B}|}{|\mathbf{B}|}\right)^2\right) \\ &\approx \Omega_0 + (y - y_0){\Omega_0}'.\end{aligned} \tag{3.3.6}$$

Substituting the Taylor expansion of $\Omega(y)$, denoted in equation (3.3.6), into the ODEs in equations (3.3.4) and (3.3.5), then expanding squares and neglecting second-order terms the ODEs become,

$$\ddot{x} + \Omega_0^2\dot{x} \approx -2\Omega_0{\Omega_0}'(y - y_0)\dot{x} + {\Omega_0}'\dot{y}^2 \tag{3.3.7}$$

$$\ddot{y} + \Omega_0^2\dot{y} \approx -2\Omega_0{\Omega_0}'(y - y_0)\dot{x} - {\Omega_0}'\dot{x}\dot{y}. \tag{3.3.8}$$

It is possible to consider $\dot{x}$ and $\dot{y}$ as a combination of their zeroth-order solution (terms on the left side of the above equations) and small perturbations that are first-order (terms on the right side of the above equations),

$$\dot{x} = \dot{x}_0 + \dot{x}_1 \tag{3.3.9}$$

$$\dot{y} = \dot{y}_0 + \dot{y}_1, \tag{3.3.10}$$

where $\dot{x}_0$ and $\dot{y}_0$ are the zeroth-order solution ($O(1)$), and $\dot{x}_1$ and $\dot{y}_1$ are the small perturbations of order $O\left(\frac{|\delta\mathbf{B}|}{|\mathbf{B}|}\right)$. Now substitute the new expressions for $\dot{x}$ and $\dot{y}$, denoted in equations (3.3.9) and (3.3.10), into equations (3.3.7) and (3.3.8):

$$\begin{aligned}(\ddot{x}_0 + \ddot{x}_1) + \Omega_0^2(\dot{x}_0 + \dot{x}_1) = &-2\Omega_0{\Omega_0}'(y - y_0)(\dot{x}_0 + \dot{x}_1) \\ &+ {\Omega_0}'\left(\dot{y}_0^2 + 2\dot{y}_0\dot{y}_1 + \dot{y}_1^2\right)\end{aligned} \tag{3.3.11}$$

$$\begin{aligned}(\ddot{y}_0 + \ddot{y}_1) + \Omega_0^2(\dot{y}_0 + \dot{y}_1) = &-2\Omega_0{\Omega_0}'(y - y_0)(\dot{y}_0 + \dot{y}_1) \\ &+ {\Omega_0}'\left(\dot{x}_0\dot{y}_0 + \dot{x}_0\dot{y}_1 + \dot{x}_1\dot{y}_0 + \dot{x}_1\dot{y}_1\right).\end{aligned} \tag{3.3.12}$$

Neglecting terms of second-order ($O\left(\left(\frac{|\delta\mathbf{B}|}{|\mathbf{B}|}\right)^2\right)$), equations (3.3.11) and (3.3.12) become

$$(\ddot{x}_0 + \ddot{x}_1) + \Omega_0^2(\dot{x}_0 + \dot{x}_1) = -2\Omega_0{\Omega_0}'(y - y_0)\dot{x}_0 + {\Omega_0}'\dot{y}_0^2 \tag{3.3.13}$$

$$(\ddot{y}_0 + \ddot{y}_1) + \Omega_0^2(\dot{y}_0 + \dot{y}_1) = -2\Omega_0{\Omega_0}'(y - y_0)\dot{y}_0 - {\Omega_0}'\dot{x}_0\dot{y}_0. \tag{3.3.14}$$

Equations (3.3.13) and (3.3.14) can be split into their zero and first-order terms as follows:

$$\ddot{x}_0 + \Omega_0^2 \dot{x}_0 = 0 \tag{3.3.15}$$

$$\ddot{y}_0 + \Omega_0^2 \dot{y}_0 = 0 \tag{3.3.16}$$

$$\ddot{x}_1 + \Omega_0^2 \dot{x}_1 = -2\Omega_0 \Omega_0{}'(y - y_0)\dot{x}_0 + \Omega_0{}' \dot{y}_0^2 \tag{3.3.17}$$

$$\ddot{y}_1 + \Omega_0^2 \dot{y}_1 = -2\Omega_0 \Omega_0{}'(y - y_0)\dot{y}_0 - \Omega_0{}' \dot{x}_0 \dot{y}_0. \tag{3.3.18}$$

We have already solved (3.3.15) and (3.3.16) in Section 3.1, hence

$$\dot{x}_0 = v_{\perp_0} \cos(\Omega_0 t) \tag{3.3.19}$$

$$\dot{y}_0 = -v_{\perp_0} \sin(\Omega_0 t). \tag{3.3.20}$$

Note that from (3.1.15)

$$(y - y_0) = \frac{v_{\perp_0}}{\Omega_0} \cos(\Omega_0 t). \tag{3.3.21}$$

We are now able to find the solutions to the first-order terms by substituting the values for $\dot{x}_0$ from equation (3.3.19), $\dot{y}_0$ from equation (3.3.20), and $(y - y_0)$ from equation (3.3.21) into the ODEs of the perturbations, denoted in equations (3.3.17) and (3.3.18):

$$\ddot{x}_1 + \Omega_0^2 \dot{x}_1 = -2\Omega_0{}' v_{\perp_0}^2 \cos^2(\Omega_0 t) + \Omega_0{}' v_{\perp_0}^2 \sin^2(\Omega_0 t) \tag{3.3.22}$$

$$\ddot{y}_1 + \Omega_0^2 \dot{y}_1 = 3\Omega_0{}' v_{\perp_0} \cos(\Omega_0 t) \sin(\Omega_0 t). \tag{3.3.23}$$

Both of these equations can be simplified further using trigonometric identities. In equation (3.3.22) use $\cos^2(x) = \frac{1}{2}(1+\cos(2x))$ and $\sin^2(x) = \frac{1}{2}(1-\cos(2x))$. In equation (3.3.23) use $\sin(2x) = 2\sin(x)\cos(x)$. Equations (3.3.22) and (3.3.23) then become;

$$\ddot{x}_1 + \Omega_0^2 \dot{x}_1 = -\frac{1}{2} v_{\perp_0}^2 \Omega_0{}' \left(1 + 3\cos(2\Omega_0 t)\right) \tag{3.3.24}$$

$$\ddot{y}_1 + \Omega_0^2 \dot{y}_1 = \frac{3}{2} v_{\perp_0}^2 \Omega_0{}' \sin(2\Omega_0 t). \tag{3.3.25}$$

These can be solved for $\dot{x}_1$ and $\dot{y}_1$ as second-order inhomogeneous differential equations. The solutions to $\dot{x}_{1_{CF}}$ and $\dot{y}_{1_{CF}}$ are the same as the zero-order solution for $\dot{x}_0$ and $\dot{y}_0$ but now include $v_{\perp_1}$ rather than $v_{\perp_0}$. The following particular integrals are used to solve $\dot{x}_{1_{PI}}$ and $\dot{y}_{1_{PI}}$:

$$\dot{x}_{1_{PI}} = A + B\cos(2\Omega_0 t)$$

$$\dot{y}_{1_{PI}} = C\sin(2\Omega_0 t).$$

Determining the constants A, B, and C. Then combining the complementary functions and particular integrals result in the following solutions for $\dot{x}_1$ and $\dot{y}_1$:

$$\dot{x}_1 = v_{\perp_1} \cos(\Omega_0 t) + \frac{1}{2} v_{\perp_0}^2 \frac{\Omega_0{}'}{\Omega_0^2} \cos(2\Omega_0 t) - v_{\perp_0}^2 \frac{\Omega_0{}'}{2\Omega_0^2} \tag{3.3.26}$$

$$\dot{y}_1 = -v_{\perp_1} \sin(\Omega_0 t) - \frac{1}{2} v_{\perp_0}^2 \frac{\Omega_0{}'}{\Omega_0^2} \sin(2\Omega_0 t). \tag{3.3.27}$$

The final solution for $\dot{x}$ and $\dot{y}$ is the sum of the zeroth and first-order solutions,

$$\dot{x} = (v_{\perp_0} + v_{\perp_1})\cos(\Omega_0 t) + \frac{1}{2}v_{\perp_0}^2 \frac{\Omega_0{}'}{\Omega_0^2}\cos(2\Omega_0 t) - v_{\perp_0}^2 \frac{\Omega_0{}'}{2\Omega_0^2} \tag{3.3.28}$$

$$\dot{y} = -(v_{\perp_0} + v_{\perp_1})\sin(\Omega_0 t) - \frac{1}{2}v_{\perp_0}^2 \frac{\Omega_0{}'}{\Omega_0^2}\sin(2\Omega_0 t) \tag{3.3.29}$$

$$\dot{z} = v_{\parallel}. \tag{3.3.30}$$

Integrating these expressions with respect to time and applying initial conditions will give the full analytical solution for the particle's trajectory.

$$x = \frac{v_{\perp_0} + v_{\perp_1}}{\Omega_0}\sin(\Omega_0 t) + \frac{1}{4}v_{\perp_0}^2 \frac{\Omega_0{}'}{\Omega_0^3}\sin(2\Omega_0 t) - v_{\perp_0}^2 \frac{\Omega_0{}'}{2\Omega_0^2} t + x_0 \tag{3.3.31}$$

$$y = \frac{v_{\perp_0} + v_{\perp_1}}{\Omega_0}\cos(\Omega_0 t) + \frac{1}{4}v_{\perp_0}^2 \frac{\Omega_0{}'}{\Omega_0^3}\cos(2\Omega_0 t) + y_0 \tag{3.3.32}$$

$$z = v_{\parallel} t + z_0 \tag{3.3.33}$$

$$\tag{3.3.34}$$

The steps to derive this perturbation solution came from private communications with Professor Duncan Mackay. From the non-oscillatory term on the RHS of equation (3.3.31) it can be seen that the particle drifts in the x-direction. This drift is perpendicular to both the applied field, $\mathbf{B}$, and the gradient of the field, ∇B, with a magnitude of

$$v_G = -v_{\perp_0}^2 \frac{\Omega_0{}'}{2\Omega_0^2}.$$

This differs from the drift seen in Section 3.2 as it now depends on the properties of the particle, rather than simply the applied fields (as in Section 3.2). This means that oppositely charged particles will drift in opposite directions. Furthermore, the mass and the charge of the particle will also affect its drift velocity. The particle's average velocity over one time period ($T = 2\pi/\Omega_0$) is

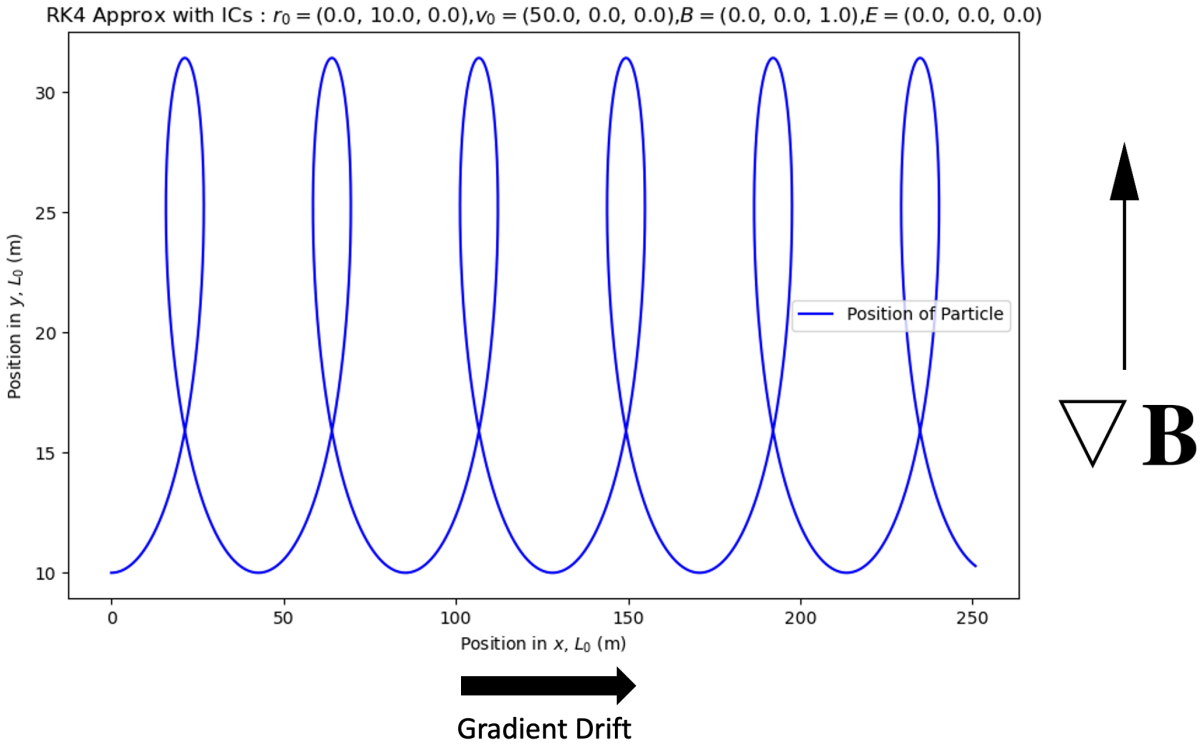
$$\langle \mathbf{v} \rangle = \left(-v_{\perp}^2 \frac{\Omega_0{}'}{2\Omega_0^2}, 0, v_{\parallel} \right),$$

therefore the average position of the particle over one time period (its guiding centre) is:

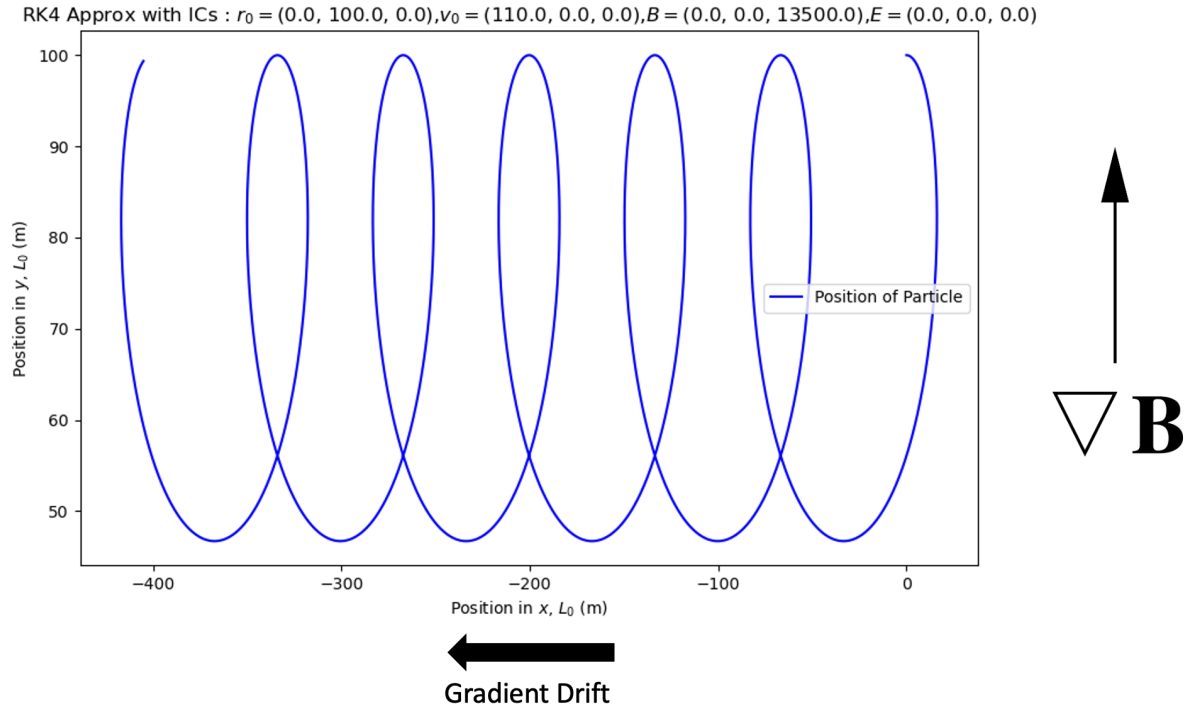
$$\langle \mathbf{r} \rangle = \left(-v_{\perp}^2 \frac{\Omega_0{}'}{2\Omega_0^2} t + x_0, y_0, v_{\parallel} t + z_0 \right).$$

The general expression for the gradient drift can be derived, although it is out of the scope of this project. The general expression is taken from Boyd and Sanderson [2003]:

$$\mathbf{v}_G = \frac{W_{\perp}\left(\mathbf{B} \times \nabla\right) B}{qB^3}.$$



(a) Drift of an electron due to an inhomogeneous magnetic field.



(b) Drift of a proton due to an inhomogeneous magnetic field.

Figure 3.7: Examples of gradient drift for positively and negatively charged particles. Note the applied fields here are unphysical and used to demonstrate the drifts on the different particles.

This drift for positively and negatively charged particles can be seen in Figure 3.7b and Figure 3.7a, respectively. Note that positively charged particles will drift in the negative $\hat{\mathbf{x}}$-direction and vice versa. This is because the particles will orbit in opposite directions, hence the drift due to the gradient in the magnetic field will be in opposite directions. The trajectories of the particles make physical sense as when in regions of stronger magnetic field (greater y) the radius of the orbit decreases, causing tighter turns at the top. The particle will also move at a faster oscillatory velocity in these regions as the gyro-frequency, $\Omega(y)$, is larger in these regions. Hence the the period of the oscillation decreases. This can be related to the first configuration of electromagnetic fields that we considered; the constant magnetic field. The particle would orbit faster and with a smaller radius when in a stronger magnetic field as illustrated in Figure 3.1. It should be noted that the fields applied to the particles in Figure 3.7 do not satisfy the assumptions made at the start of this section that $|\delta\mathbf{B}| \ll |\mathbf{B}|$. In order to get the orbits seen in Figure 3.7 the field applied field had to have a large inhomogeneity - resulting in the visible difference in orbital radius. Under the assumption the inhomogeneity is very small the analytical perturbation solution leads to an orbit resembling that seen in Figure 3.8.

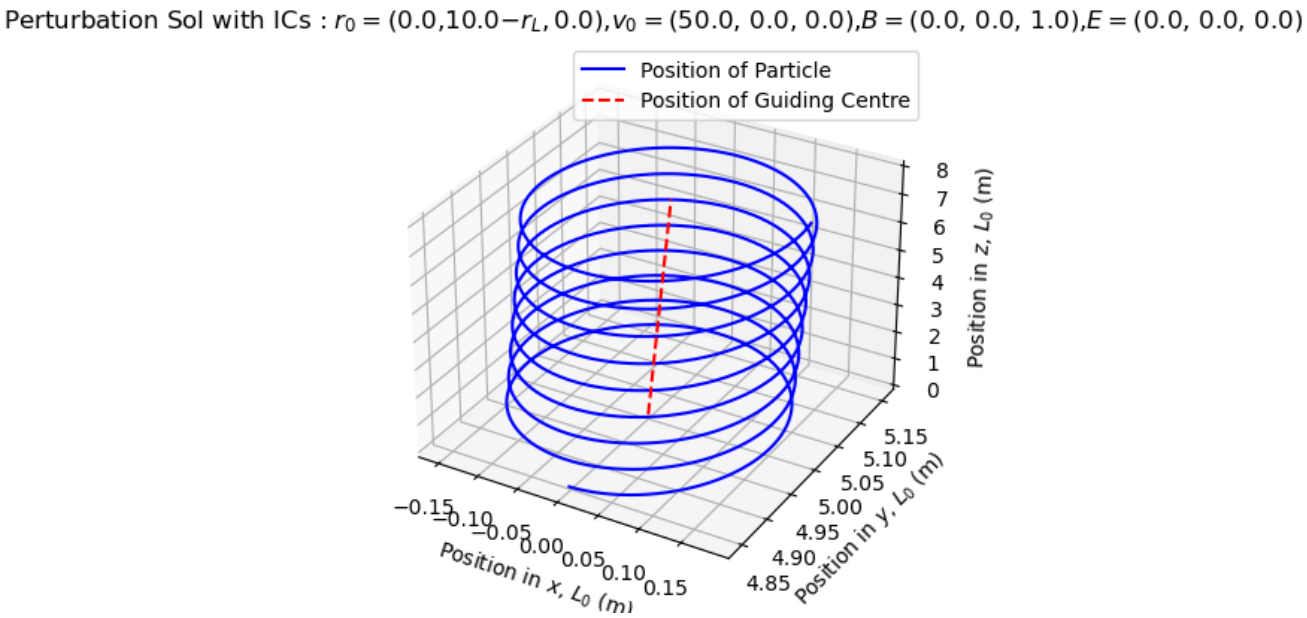


Figure 3.8: Orbit of an electron in an inhomogeneous magnetic field, satisfying the assumptions to solve for a perturbation solution. $B_z = 0.3y + 5$.

Note in order to computationally implement the perturbation solution we set $v_{\perp_1} = 0$. Doing so results in the derived solution matching the analytical solution provided by Boyd and Sanderson [2003] which also excludes second-order terms.

Due to the omission of the second-order terms, the error analysis will be different from what has been seen before as the analytical perturbation solution is not completely accurate. It is possible to approximately calculate how accurate equations (3.3.31) and (3.3.32) will be by considering the magnitude of $\left(\frac{|\delta\mathbf{B}|}{|\mathbf{B}|}\right)^2$. Since the perturbation solution does not contain terms of this order, it is an upper limit of how accurate the solution can be. For the initial conditions in Figure 3.8 the magnitude of $\left(\frac{|\delta\mathbf{B}|}{|\mathbf{B}|}\right)^2$ is of order 10^{-6}.It can be seen in Figure 3.9 that the accuracy of the solution converges to a value of the predicted magnitude as expected.

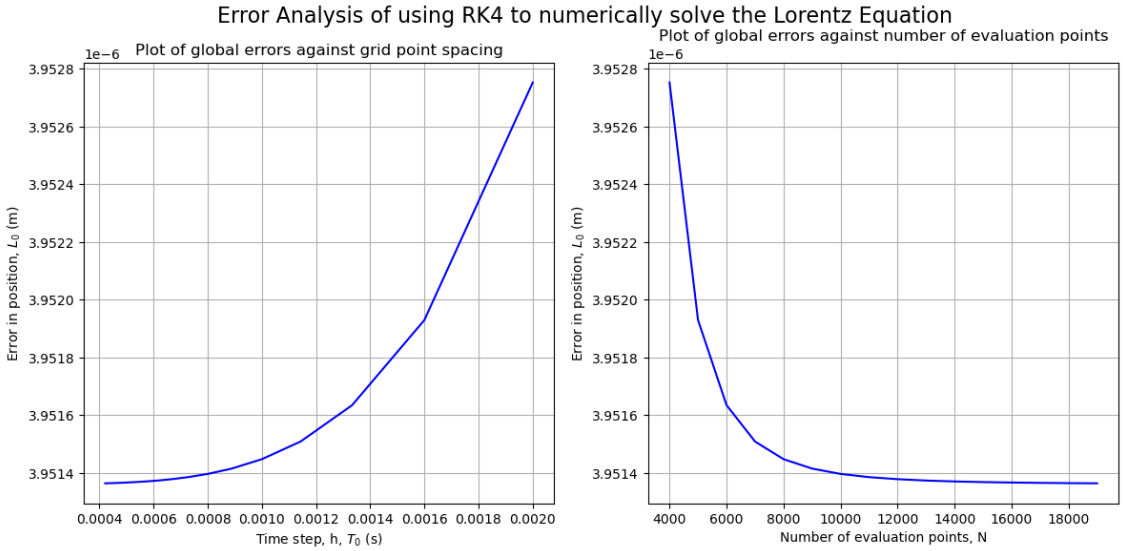


Figure 3.9: Error analysis of RK4 method for values of h from $\sim$0.002 to $\sim$0.0004 and values of N from 4000 to 20000 for a particle and fields with initial conditions consistent with that seen in Figure3.8.

3.4 Curvature Drift in an Inhomogeneous Magnetic Field

The Earth's magnetic field lines are not straight, as we have assumed throughout this project, they are curved as can be seen in the image of Earth's magnetosphere in Figure 1.1. The curvature of field lines gives rise to a new drift velocity. For example, if the applied fields are $\mathbf{B} = (0, B_y(z), B_z)$ (see Figure 3.10) and $\mathbf{E} = \mathbf{0}$ then (1.1.3)-(1.1.5) become:

$$\ddot{x} = \Omega\dot{y} - \Omega_y(z)\dot{z} \tag{3.4.1}$$

$$\ddot{y} = -\Omega\dot{x} \tag{3.4.2}$$

$$\ddot{z} = \Omega_y(z)\dot{x}, \tag{3.4.3}$$

where $\Omega_y(z) = \frac{q}{m}B_y(z)$. We will assume both $B_y(z)$ and $\frac{\partial B_y(z)}{\partial z}$ are small quantities. Hence, Ω_y is also a small quantity. It is possible to take them as small quantities as in a real-world setting the field curvature is extremely small relative to the distance travelled along the field in a gyro-period. It is possible to write equations (3.4.1) and (3.4.2) in a form that resembles the ODEs we've considered previously by differentiating them with respect to time and neglecting the squares of small terms. In doing so the equations of motion become:

$$\dddot{x} + \Omega^2\dot{x} = -\Omega_y{}'(z)\dot{z}^2 \tag{3.4.4}$$

$$\dddot{y} + \Omega^2\dot{y} = \Omega\Omega_y(z)\dot{z} \tag{3.4.5}$$

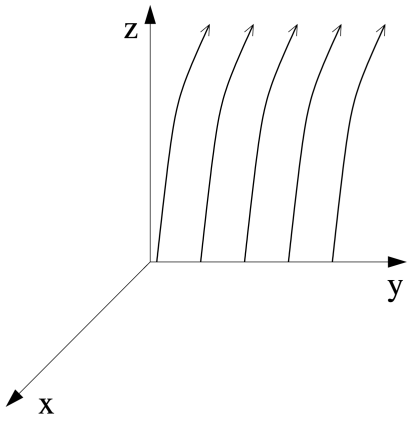


Figure 3.10: Curved Magnetic field lines where the curvature does not depend on x

This system of equations is still coupled. However, the LHS resembles what we have considered in the previous sections, which when solved results in oscillatory motion in the xy-plane. Both are inhomogeneous differential equations, thus in addition to the oscillations, there will be drift in the $\hat{\mathbf{x}}$-direction and in the $\hat{\mathbf{y}}$-direction. The drift in the $\hat{\mathbf{x}}$-direction is known as curvature drift and is due to the curvature of the magnetic field. In other words, it is due to the inhomogeneous magnetic field in the $\hat{\mathbf{y}}$-direction, this explains why equation (3.4.4) bears such a close resemblance to the Gradient Drift Equation (3.3.4). The drift in the $\hat{\mathbf{y}}$-direction simply keeps the guiding centre of the orbit moving along the curved magnetic field lines in the yz-plane as seen in Figure 3.10. As the equations are still coupled it is not possible to derive a full analytical solution for the position of the particle as we have done before. We are able to determine the drifts of the particle by considering the small quantities ($B_y(z)$ and $\frac{\partial B_y(z)}{\partial z}$) and the average velocity of the particle after one Larmor orbit (with frequency Ω). We want the RHS of equations (3.4.4) and (3.4.5) to be approximately constant so the particular integrals of the inhomogeneous ODEs can be considered. This is firstly achieved by $\Omega_y(z)$ and $\Omega_y{}'$ both being very small quantities, thus any changes in them can be ignored. Also, $\ddot{z}$ is a small quantity (from equation (3.4.3)), meaning $\dot{z}$ will be approximately constant. As the RHS of the inhomogeneous ODEs in equations (3.4.4) and (3.4.5) are approximately constant the particular integrals can be solved and the drifts can be determined.

$$v_c = \langle \dot{x} \rangle = -\frac{\Omega_y{}'}{\Omega^2} v_\parallel^2 \tag{3.4.6}$$

$$\langle \dot{y} \rangle = \frac{\Omega_y}{\Omega} v_\parallel \tag{3.4.7}$$

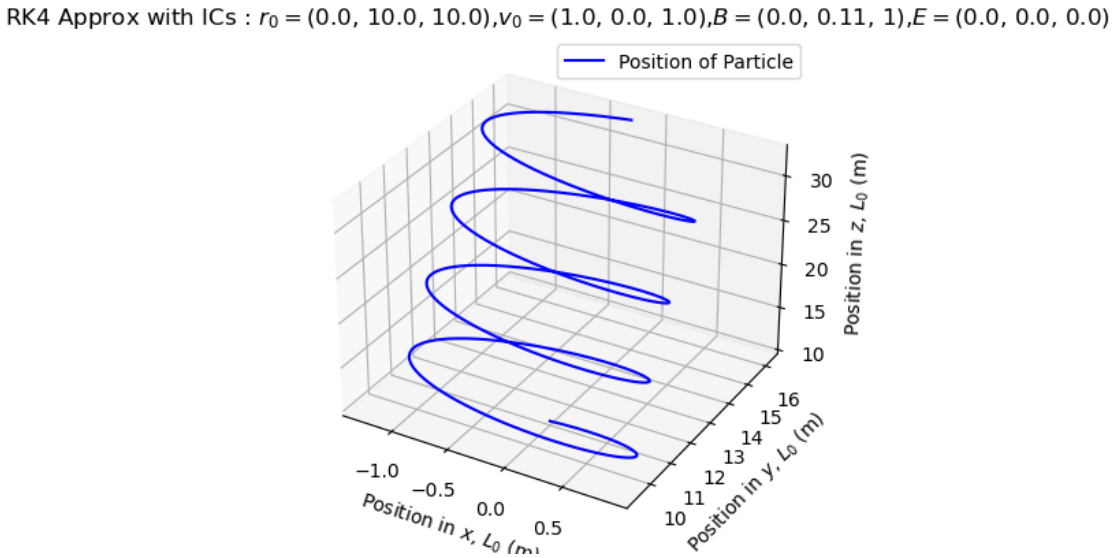
Where v_c is the curvature drift and $v_\parallel$ is the constant velocity parallel to the magnetic field in the $\hat{\mathbf{z}}$-direction. The derivation of these drifts came from private communication with Professor Duncan Mackay.
The general form of the drift due to the curvature of the field may be written as

$$\mathbf{v}_c = 2W_\parallel \left(\frac{\mathbf{B} \times (\mathbf{B} \cdot \nabla)\, \mathbf{B}}{qB^4} \right), \tag{3.4.8}$$

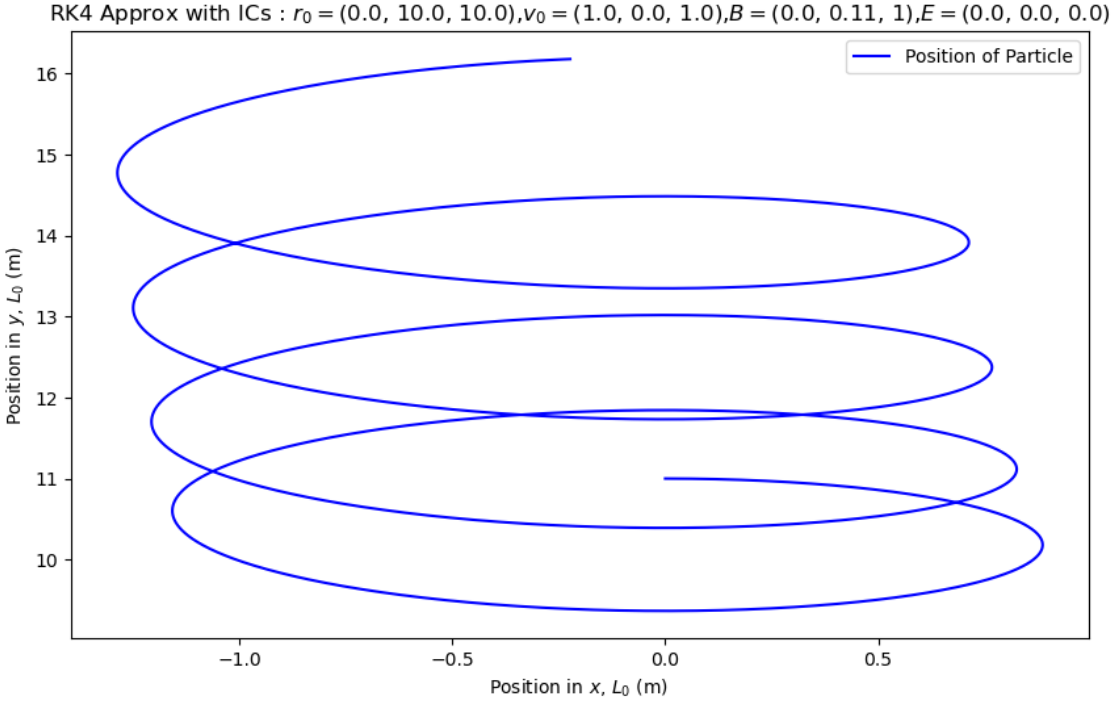
where $W_\parallel$ is the kinetic energy of the particle in the $\hat{\mathbf{z}}$-direction [Boyd and Sanderson, 2003]. The derivation of this expression is outside of the scope of this project. By considering the drift in the $\hat{\mathbf{y}}$-direction it is possible to see that this drift keeps the guiding centre moving along a magnetic field line in the yz-plane:

$$\frac{\langle \dot{y} \rangle}{\langle \dot{z} \rangle} = \frac{v_\parallel \Omega_y / \Omega}{v_\parallel} = \frac{B_y}{B_z}. \tag{3.4.9}$$

An example of this drift can be seen in Figure 3.11. The drift can be seen best in the 2D depiction in Figure 3.11b, where the drift in the $\hat{\mathbf{y}}$-direction (keeping the electron orbiting a field line) is more prominent and the curvature drift in the $\hat{\mathbf{x}}$-direction is small in comparison.



(a) 3D depiction of an electron's orbit in a curved magnetic field



(b) 2D depiction of an electron's orbit in a curved magnetic field.

Figure 3.11: Curvature drift of an electron.

4 Conclusion

Throughout this project, we have considered four different configurations of electromagnetic fields and modelled the dynamics of each case. All the configurations are somewhat simplified in comparison to the electromagnetic fields found in the magnetosphere. However, what has been considered in this project is a strong starting point in understanding the individual dynamics of charged particles in the magnetosphere. The applications of what has been considered is wide-ranging. As discussed, the biggest application is the safety of spacecraft.

Where possible a full analytical solution for the particle orbit was derived. This was done by decoupling the particle's equations of motion and solving a second-order ODE for the particle's velocity, then integrating this term to resolve the trajectory of the particle. In an inhomogeneous magnetic field, it was only possible to derive a perturbation solution. This was done by requiring the inhomogeneity to be small in comparison to the field itself. This allowed us to take the Taylor expansion of the particle's gyro-frequency about its guiding centre. Allowing the particle's orbit to be determined.

The error analysis was an integral part of this project. By deriving Euler's Method, it was possible to see the global error had first-order convergence with respect to the step size, h. Whereas, the fourth-order Runge-Kutta has a global error convergence proportional to its namesake. The expected convergence was exhibited in all the error analysis conducted. Furthermore, the expected patterns of how error changes as a function of time when modelling a sinusoidal function were exhibited. These checks confirm the implementation of the numerical methods was done correctly. Furthermore, when only a perturbation solution was able to be derived, as in Section 3.3, the RK4 method converged on an error that was the same order of magnitude as the terms neglected by the perturbation solution. In this case, the numerical method was more accurate than the derived solution. However, it was encouraging to see that the error matched what was expected due to order the of the neglected terms.

As mentioned before, the numerical solvers I created for this project are able to handle significantly more complex configurations of electromagnetic fields than the ones considered in this project. The main exclusions of this project were in the form of time varying magnetic fields and magnetic mirrors. Both of these considerations are relevant to painting an accurate picture of the electromagnetic fields found in the magnetosphere. Time varying magnetic fields (i.e. waves) can accelerate particles in the magnetosphere to relativistic energies, thus increasing the potential risk to space craft. The magnetic mirroring effect, whereby particles are reflected from strong magnetic field regions, acts to trap particles on magnetic field lines. This is an important effect in Earth's magnetosphere, where the magnetic field strength increases along field lines towards the ionospheres. This mirroring allows particles to remain in the magnetosphere on long time scales, hence permitting consistent energisation before they are lost from the system. Due to the extensive numerical side of this project, there was not enough time to consider the particle dynamics in these configurations. However, with the code I have written, they will undoubtedly be the first consideration for my future work in this area.

A Python Code

For completeness all code used throughout this project will be included here, we will start with the already seen numerical solvers;

```python
# Define function that will give the Acceleration due to Lorentz Force
def lorentz_acc(x, y, z, v_x, v_y, v_z, t):
        # Calculate the components of the Lorentz force (electric + magnetic)
        ax = q/m * (Ex(x,y,z,t) + ((v_y * Bz(x,y,z,t)) - (v_z * By(x,y,z,t))))
        ay = q/m * (Ey(x,y,z,t) + ((v_z * Bx(x,y,z,t)) - (v_x * Bz(x,y,z,t))))
        az = q/m * (Ez(x,y,z,t) + ((v_x * By(x,y,z,t)) - (v_y * Bx(x,y,z,t))))
        return ax, ay, az
```

```python
# Define function that will give the Acceleration due to Lorentz Force
def lorentz_acc(x, y, z, v_x, v_y, v_z, t):
        # Calculate the components of the Lorentz force (electric + magnetic)
        ax = q/m * (Ex(x,y,z,t) + ((v_y * Bz(x,y,z,t)) - (v_z * By(x,y,z,t))))
        ay = q/m * (Ey(x,y,z,t) + ((v_z * Bx(x,y,z,t)) - (v_x * Bz(x,y,z,t))))
        az = q/m * (Ez(x,y,z,t) + ((v_x * By(x,y,z,t)) - (v_y * Bx(x,y,z,t))))
        return ax, ay, az

def euler_approx(N,x_0,y_0,z_0,v_x,v_y,v_z,B_x,B_y,B_z,E_x,E_y,E_z,q,m,t_end):
    # Calculate step size
    h = (t_end)/(N-1)

    # Assigning the first value of each component in x,y, and z to match IC
    x = x_0
    y = y_0
    z = z_0

    # Stores for position of particle
    xvals = [x]
    yvals = [y]
    zvals = [z]

    t = 0

    for i in range(N-1):
        # Components of acceleration at time t for the current position
        a_x,a_y,a_z = lorentz_acc(x,y,z,v_x,v_y,v_z,t)
        # Update position using its derivative (velocity)
        x += (v_x * h)
        y += (v_y * h)
        z += (v_z * h)
        # Update velocity using its derivative (acceleration)
        v_x += (a_x * h)
        v_y += (a_y * h)
        v_z += (a_z * h)
        xvals.append(x) # appending a list of x for plot
        yvals.append(y)
        zvals.append(z)
        # Update time by h
        t += h

    # Convert lists to numpy arrays for convenience
    xvals = np.array(xvals)
```

```
    yvals = np.array(yvals)
    zvals = np.array(zvals)
    return xvals, yvals, zvals
```

```
def rk4_approx(N, x_0, y_0, z_0, v_x, v_y, v_z, B_x, B_y, B_z, E_x, E_y, E_z, q, m, t_end):
    # Calculate step size
    h = t_end / (N - 1)

    # Assigning the first value of each component in x,y, and z to match IC
    x = x_0
    y = y_0
    z = z_0

    # Stores for position of particle
    xvals = [x]
    yvals = [y]
    zvals = [z]

    t = 0

    for i in range(N-1):
        # Current accelerations
        a_x, a_y, a_z = lorentz_acc(x,y,z,v_x,v_y,v_z,t)

        # K1
        k1_vx, k1_vy, k1_vz = h*a_x, h*a_y, h*a_z
        k1_x, k1_y, k1_z = h*v_x, h*v_y, h*v_z

        # K2
        # Update the acceleration from Lorentz Force
        ax_mid1, ay_mid1, az_mid1 = lorentz_acc(    # Linebreak due to long function input
            x+(0.5*k1_x),y+(0.5*k1_y),z+(0.5*k1_z),
            v_x+(0.5*k1_vx),v_y+(0.5*k1_vy),v_z+(0.5*k1_vz),t+0.5*h)

        k2_vx, k2_vy, k2_vz = h*ax_mid1, h*ay_mid1, h*az_mid1
        k2_x, k2_y, k2_z = h*(v_x + 0.5*k1_vx), h*(v_y + 0.5*k1_vy), h*(v_z + 0.5*k1_vz)

        # K3
        # Update the acceleration from Lorentz Force
        ax_mid2, ay_mid2, az_mid2 = lorentz_acc(
            x+(0.5*k2_x),y+(0.5*k2_y),z+(0.5*k2_z),
            v_x+(0.5*k2_vx),v_y+(0.5*k2_vy),v_z+(0.5*k2_vz),t+0.5*h)

        k3_vx, k3_vy, k3_vz = h * ax_mid2, h * ay_mid2, h * az_mid2
        k3_x, k3_y, k3_z = h * (v_x + 0.5*k2_vx), h * (v_y + 0.5*k2_vy), h * (v_z + 0.5*k2_vz)

        # K4
        # Update the acceleration from Lorentz Force
        ax_end, ay_end, az_end = lorentz_acc(
            x+(k3_x),y+(k3_y),z+(k3_z),v_x+(k3_vx),v_y+(k3_vy),v_z+(k3_vz),t+h)

        k4_vx, k4_vy, k4_vz = h * ax_end, h * ay_end, h * az_end
        k4_x, k4_y, k4_z = h * (v_x + k3_vx), h * (v_y + k3_vy), h * (v_z + k3_vz)

        # Update velocity and position
        v_x += (k1_vx + (2*k2_vx) + (2*k3_vx) + k4_vx) / 6
```

```
        v_y += (k1_vy + (2*k2_vy) + (2*k3_vy) + k4_vy) / 6
        v_z += (k1_vz + (2*k2_vz) + (2*k3_vz) + k4_vz) / 6

        x += (k1_x + (2*k2_x) + (2*k3_x) + k4_x) / 6
        y += (k1_y + (2*k2_y) + (2*k3_y) + k4_y) / 6
        z += (k1_z + (2*k2_z) + (2*k3_z) + k4_z) / 6

        # Append the new values to the lists
        xvals.append(x)
        yvals.append(y)
        zvals.append(z)

        t += h

    # Convert lists to numpy arrays for convenience
    xvals = np.array(xvals)
    yvals = np.array(yvals)
    zvals = np.array(zvals)

    return xvals, yvals, zvals
```

Now the functions that represent the analytical and perturbation solution will be included;

```
# Analytical solution for constant magnetic field
def x_exact1(x0, vx, vy, Omega, t):
    v_perp = np.sqrt(v_x**2 + v_y**2)
    x =  (v_perp/Omega) * np.sin(Omega * t) + x0
    return x

def y_exact1(y0, vx, vy, Omega, t):
    v_perp = np.sqrt(v_x**2 + v_y**2)
    y = (v_perp/Omega) * np.cos(Omega * t) + y0
    return y

def z_exact(z0, vz, t):
    z = vz * t + z0
    return z

# Analytical solution for constant electric and magnetic fields
def x_exact2(x0,vx,vy,Omega,t):
    return (vx/Omega - E_y/(B_z*Omega))*np.sin(Omega*t) + (E_y/B_z)*t + x0

def y_exact2(y0,vx,vy,Omega,t):
    return (vx/Omega - E_y/(B_z*Omega))*np.cos(Omega*t)  + y0

# Function for z is the same as previous case

# Perturbation Solution for inhomogeneous magnetic field
Omega_0 = (q/m)*(0.3*y_0 + 5) # Change depending on the chosen B_z
Omega_dash = (q/m)*0.3

def x_exact3(x0, vx, vy, Omega, t):
    v_perp = np.sqrt(v_x**2 + v_y**2)
    x =  (v_perp/Omega)*np.sin(Omega * t) + ((Omega_dash * v_perp**2)/(4*Omega**3))*np.sin(2*Omega*t) \
         - (Omega_dash * v_perp**2 *t)/(2*Omega**2) + x0
    return x

```

```
def y_exact3(y0, vx, vy, Omega, t):
    v_perp = np.sqrt(v_x**2 + v_y**2)
    y =  (v_perp/Omega)*np.cos(Omega * t) + ((Omega_dash * v_perp**2)/(4*Omega**3))*np.cos(2*Omega*t)+ y0
    return y

# Function for z is the same as previous cases

# Solver defined to quickly return arrays of particle position
def solver(xexact,yexact,zexact,N,x_0,y_0,z_0,v_x,v_y,v_z,B_x,B_y,B_z,E_x,E_y,E_z,q,m,t_max):
    tvals = np.linspace(0,t_max,N)
    Omega = (q*B_z)/m
    x = xexact(x_0,v_x,v_y,Omega,tvals)
    y = yexact(y_0,v_x,v_y,Omega,tvals)
    z = zexact(z_0,v_z,tvals)
    return x,y,z
```

Furthermore, when possible, the functions representing the movement of the guiding centre were also coded up;

```
# Guiding centre in a constant magnetic field
def gc_1(x0,y0,z0,vx,vz,t):
    x = x0 +t -t
    y = y0 +t -t
    z = vz*t +z0
    return x,y,z

# Guiding centre in constant electric and magnetic fields
def gc_2(x0,y0,z0,vz,E_y,B_z,t):
    x = (E_y/B_z)*t +x0
    y = y0 +t -t
    z = vz*t +z0
    return x,y,z

# Guiding centre in a inhomogeneous magnetic field
def gc_3(x0,y0,z0,vx,vz,Omega_0,Omega_dash,t):
    x = -(Omega_dash * v_x**2 *t)/(2*Omega_0**2) + x0
    y = y0 +t-t
    z = vz*t + z0
    return x,y,z
```

In order to run these functions, initial conditions of the particles motion must be defined. The functions of the electric and magnetic fields must be defined and then initial values taken from these functions. This code was updated for each configuration of applied fields and is shown below;

```
## Initial Conditions

# Define functions for each component of Magnetic and Electric Field (update accordingly)
def Bx(x,y,z,t):
    return 0.0
def By(x,y,z,t):
    return 0.0
def Bz(x,y,z,t):
    return 0.0
def Ex(x,y,z,t):
    return 0.0
def Ey(x,y,z,t):
    return 0.0
def Ez(x,y,z,t):
```

```
    return 0.0

# Initial position
x_0,y_0,z_0 = 0.0,0.0,0.0
# Components of initial velocity
v_x,v_y,v_z = 1.0,0.0,1.0

# Components of initial Electric Field
E_x, E_y,E_z = Ex(x_0,y_0,z_0,0),Ey(x_0,y_0,z_0,0),Ez(x_0,y_0,z_0,0)
# Components of initial Magnetic field
B_x,B_y,B_z = Bx(x_0,y_0,z_0,0),By(x_0,y_0,z_0,0),Bz(x_0,y_0,z_0,0)

m = 1.0 # Mass of the particle
q = -1.0 # Charge

Omega = (q/m)*B_z

# Chose a value of N and t to run functions over
N = 10000
t_max = 10
```

Finally the Python code that was used to calculate the error in the particle's position is shown below;

```
# Error as a function of N and h
Nvals = np.arange(1000,20000,1000)
Err_vals_e = []
Err_vals_rk = []
hvals = []

for N in Nvals:
    tvals = np.linspace(0,t_max,N)

    h = t_max/(N-1)
    hvals.append(h)

    x1,y1,z1=solver(x_exact,y_exact,z_exact,N,x_0,y_0,z_0,v_x,v_y,v_z,B_x,B_y,B_z,E_x,E_y,E_z,q,m,t_max)
    x2,y2,z3=euler_approx(N,x_0,y_0,z_0,v_x,v_y,v_z,B_x,B_y,B_z,E_x,E_y,E_z,q,m,t_max)
    x3,y3,z3=rk4_approx(N,x_0,y_0,z_0,v_x,v_y,v_z,B_x,B_y,B_z,E_x,E_y,E_z,q,m,t_max)

    # Maximum error will be the final value as Global
    err_euler = np.sqrt((x2[-1]-x1[-1])**2 + (y2[-1]-y1[-1])**2 + (z2[-1]-z1[-1])**2)
    err_rk4 = np.sqrt((x3[-1]-x1[-1])**2 + (y3[-1]-y1[-1])**2 + (z3[-1]-z1[-1])**2)

    Err_vals_e.append(err_euler)
    Err_vals_rk.append(err_rk4)

# Error as a function of time
N = 5000 # Sufficent N so both schemes are accurate
tvals = np.linspace(0,t_max,N)

# Use solvers to obtain arrays of position
x1,y1,z1 = solver(x_exact1,y_exact1,z_exact,N,x_0,y_0,z_0,v_x,v_y,v_z,B_x,B_y,B_z,E_x,E_y,E_z,q,m,t_max)
x2,y2,z2 = euler_approx(N,x_0,y_0,z_0,v_x,v_y,v_z,B_x,B_y,B_z,E_x,E_y,E_z,q,m,t_max)
x3,y3,z3 = rk4_approx(N,x_0,y_0,z_0,v_x,v_y,v_z,B_x,B_y,B_z,E_x,E_y,E_z,q,m,t_max)

# Find error in position for Euler and RK4
error_euler = np.sqrt((x2-x1)**2 + (y2-y1)**2 + (z2-z1)**2)
```

```
error_rk4 = np.sqrt((x3-x1)**2 + (y3-y1)**2 + (z3-z1)**2)

# Find error in x, y, and z for Euler and RK4
error_xe = abs(x2-x1)
error_ye = abs(y2-y1)
error_ze = abs(z2-z1)
error_xrk = abs(x3-x1)
error_yrk = abs(y3-y1)
error_zrk = abs(z3-z1)
```

References

J.E. Borovsky and J.A Valdivia. The earth's magnetosphere: A systems science overview and assessment. *Surveys in Geophysics*, pages 817–859, 2018. ISSN 1573-0956. doi: https://doi.org/10.1007/s10712-018-9487-x. URL `https://rdcu.be/dC8GB`. 6

T.J.M Boyd and J.J Sanderson. *Particle Orbit Theory*, chapter 2, pages 12–22. Cambridge University Press, 2003. ISBN 9780511755750. doi: 10.1017/CBO9780511755750. URL `https://www.cambridge.org/core/books/abs/physics-of-plasmas/frontmatter/09A6822EFDB1384A56663EB5F4C85244`. 4, 31, 33, 35

A. W. Degeling, R. Rankin, K. Kabin, R. Marchand, and I. R. Mann. The effect of ulf compressional modes and field line resonances on relativistic electron dynamics. *Planetary and Space Science*, 55(6):731–742, 2007. doi: 10.1016/j.pss.2006.04.039. 12

J. P. Eastwood, E. Biffis, M. A. Hapgood, L. Green, M. M. Bisi, R. D. Bentley, R. Wicks, L.-A. McKinnell, M. Gibbs, and C. Burnett. The economic impact of space weather: Where do we stand? *Risk Analysis*, 37 (2):206–218, 2017. doi: https://doi.org/10.1111/risa.12765. URL `https://onlinelibrary.wiley.com/doi/abs/10.1111/risa.12765`. 5

L.J. Lanzerotti, C. Breglia, D.W. Maurer, G.K. Johnson, and C.G. Maclennan. Studies of spacecraft charging on a geosynchronous telecommunications satellite. *Advances in Space Research*, 22(1):79–82, 1998. ISSN 0273-1177. doi: https://doi.org/10.1016/S0273-1177(97)01104-6. URL `https://www.sciencedirect.com/science/article/pii/S0273117797011046`. Solar-Terrestrial Relations: Predicting the Effects on the Near-Earth Environment. 5

A.R Lozinski, R.B Horne, S.A Glauert, G. Del Zanna, and D. Heynderickx. Solar cell degradation due to proton belt enhancements during electric orbit raising to geo. *Space Weather*, 17(7):1059–1072, 2019. doi: https://doi.org/10.1029/2019SW002213. URL `https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2019SW002213`. 5

J. H. Shue, J. K. Chao, H. C. Fu, C. T. Russell, P. Song, K. K. Khurana, and H. J. Singer. A new functional form to study the solar wind control of the magnetopause size and shape. *Journal of Geophysical Research*, 102(A5):9497–9512, May 1997. doi: 10.1029/97JA00196. 5

Daniel C. Wilkinson. National oceanic and atmospheric administration's spacecraft anomaly data base and examples of solar activity affecting spacecraft. *Journal of Spacecraft and Rockets*, 31(2):160–165, 1994. doi: 10.2514/3.26417. URL `https://doi.org/10.2514/3.26417`. 5