PHYS:5905 Homework #5a

Please submit your solutions as a single PDF file with answers to the questions asked.
Please complete required problems before lecture on Thursday, February 21, 2019.

1. (Required) **Small-Angle Collision Scattering Routine**

   (a) Write a routine the performs elastic small-angle collisions of angle θ, normally distributed with a small-angle variance $\sigma(\theta)$.

   (b) This routine will take an initial velocity vector $\mathbf{v}_i$ and scatter it away from that angle by a small θ with a random azimuthal angle ϕ about the original vector. Let us take our initial velocity vector to be $\mathbf{v}'_i = v'_0\hat{\mathbf{x}}$, where $v'_0 = 1$.

   (c) To accomplish this small-angle scattering, use the following steps:

      i. First, we rotate the velocity vector $\mathbf{v}_i$ to the z-axis, here denoted $\mathbf{v}_{inc}$. This can be done by computing the direction of the velocity $\mathbf{v}_i$ in spherical coordinates using the Cartesian to Spherical coordinate function,
      `[phi1,elev1,vmag]=cart2sph(y0(4),y0(5),y0(6));`
      where $\mathbf{y}_0 =$`[ x y z vx vy vz]` is row vector with the initial position and velocity, the azimuthal angle `phi1` ranges over $[-\pi, \pi]$ and the elevation `elev1` ranges over $[-\pi/2, \pi/2]$. If we rotate our vector about z by `-phi1` and then about y by minus the polar angle $-\theta_1 =$`elev1`$-\pi/2$, then our vector will be aligned with z. (Verify this before moving on.)

      ii. Next, we obtain a scattering through an angle θ by rotating first about the y-axis by the angle θ using a rotation matrix

      $$R_1 = \begin{pmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{pmatrix} \tag{1}$$

      We also need to allow for a rotation by ϕ about the z-axis using

      $$R_2 = \begin{pmatrix} \cos(\phi) & -\sin(\phi) & 0 \\ \sin(\phi) & \cos(\phi) & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2}$$

      Using these rotation matrices, the scattered vector $\mathbf{v}_{scat} = R_2 R_1 \mathbf{v}_{inc}$
      NOTE: The usual convention for rotation matrices is for a rotation of the *coordinate system* by angle ϕ. In a fixed coordinate system, the same rotation matrix will yield rotation of the vector by angle $-\phi$, which is why the negative signs in the matrices above may be opposite from where you are used to seeing them.

      iii. We can model small angle collisions by using normally distributed random numbers, using the Matlab function `randn`, to determine the angle θ with some small typical variance $\sigma(\theta)$ of the distribution. Thus, you can use
      `the=randn*sigmatheta;`
      to compute θ over a range with a variance $\sigma(\theta) = \pi/18$.

      iv. The azimuthal angle ϕ of the scattering is a randomly selected value with uniform probablity in the range $0 \le \phi \le 2\pi$, using
      `phi=rand*2*pi;`

      v. Finally, we perform the reverse steps of the original rotation, first $-\theta_1$ about y followed by +`phi1` about z.

   (d) (Return) Perform the scattering $n_{coll} = 1000$ times, plotting in 3D using `plot3` the track of the tip of the velocity vector (v_x, v_y, v_z) at each scattering for $\sigma(\theta) = \pi/18$. Use
   `axis([-1.  1.  -1.  1.  -1.  1.]);`
   to ensure that the entire unit sphere is included in the plot dimensions.

   (e) (Return) Produce another 3D plot using a smaller scattering angle variance, $\sigma(\theta) = \pi/180$.

2. (Required) **Implementation of Monte Carlo Collisions**

   (a) For electron single-particle motion and confinement in a magnetic mirror configuration, we will implement a collision model for electron-ion collisions with $m_i \gg m_e$, such that the electrons exchange no energy with the massive ions, but rather simply have a change in the direction of the velocity.

   (b) We will take a velocity-independent collision operator with a small collision frequency ν such that $\nu/\Omega \ll 1$. In this limit, collisions are a Poisson process with a probability for a collision in a timestep Δt given by
   $$P_{coll} = \nu \Delta t \ll 1 \tag{3}$$

   (c) Because a collision in this limit changes the velocity of the particle discontinuously (while conserving kinetic energy), the implementation of a higher order stepping scheme that depends on derivatives at previous timesteps is problematic. (Can you discern what the problem may be?). Therefore, I recommend you use Runge-Kutta timestepping since it depends only on the current timestep values to advance the particles.

   (d) Since the probability of a collision during one timestep is low, $\nu\Delta t \ll 1$, we can achieve significant computational savings by only checking to see if a collision occurs at regular intervals T (which may be longer than a single timestep $T > \Delta t$, as long as $\nu T \ll 1$).

   (e) You are free to implement your Monte Carlo Collision operator as you see fit, but I can recommend this approach:

      i. Integrate over a short interval of time T using the RK45 Adaptive Timestepping algorithm using a relative tolerance
      `options = odeset('RelTol',1.0e-3);`

      ii. At the end of each interval T, check to see if the particle has escaped the magnetic mirror (with $|z| > L/2$, where L is the length between field maxima along the mirror axis). If the particle has escaped, save the time at which the particle escaped as τ and exit the time integration of the particle motion. This time τ is a measure of the confinement time in the magnetic mirror.

      iii. At the end of each interval T, check to see if a collision has occurred.

      iv. If the collision has occurred, rotate the velocity vector of the particle by a small angle collision with variance $\sigma(\theta) = \pi/18$ (use results of previous problem).

      v. Integrate over the next short interval of time T using the RK45 Adaptive Timestepping algorithm, and repeat the steps above.

      vi. Stop the time integration at T_{end} if the particle has not yet escaped.

   (f) Set up a magnetic mirror with the parameters: mirror length $L' = 10$, minimum axial field value $B'_{00} = 10$ and mirror ratio $R_m = 4$.

   (g) Choose an initial position $\mathbf{x}_0/r_L = (0, 0, 0)$, and an initial velocity $\mathbf{v}_0/v_\perp = (0, 1, 1)$ for the electron for each run of the ensemble. You may want to limit the total integration time for each run to $\Omega T_{end} = 100\pi$ and choose the collision check interval $T = T_{end}/5000$.

   (h) Run an ensemble of runs with different collisionalities over the range $0.001 \leq \nu/\Omega \leq 1$ with at least 4 different collisionalities to sample the range. Note that, for the lowest collisionality values, the particles may not escape before T_{end}, but over a sufficiently large ensemble, you should see that the mean confinement time has a dependence on the collisionality.

   (i) For each choice of collisionality, run at least 8 single-particle motion (with Monte Carlo collisions) runs to obtain good statistics for the confinement time τ.

   (j) (Return) Produce a plot of mean confinement time $\langle\tau\rangle$ with error bars giving the standard deviation $\sigma(\tau)$ vs. the normalized collision frequency ν/Ω. You will likely want to use the Matlab plotting function `errorbar` to make this plot.

   (k) (Return) Produce a plot of the position of the particle in $(v_\parallel, v_\perp)$ space (relative to local mean magnetic field) at the point in time at which the particle is lost. The best way to do this is to write a function that inputs the vector `y0` and returns `vpar` and `vperp`.

   (l) (Optional) Plot the mean and standard deviation (`errorbar` plot) of the number of collisions as a function of normalized collision frequency ν/Ω.