STAR NOTE

Analytic Trajectory Model for Charged Particles Moving in a Constant Magnetic Field with Constant Momentum Loss

Lanny Ray
Department of Physics
The University of Texas at Austin
Austin, Texas 78712 USA

ABSTRACT

An analytic trajectory model for charged particles moving in a constant magnetic field with constant momentum loss is developed and applied to particle trajectories in the STAR TPC.

Introduction

For most of the particles produced in simulated ultrarelativistic nucleus + nucleus collisions at RHIC energies near mid-rapidity the fractional energy loss suffered during traversal of the TPC is only a few percent. However, for low p_T spiraling particles the present TPC tracking code breaks the tracks into several fragments. Successful joining of these track fragments may require a trajectory model which includes average momentum loss effects. Although energy loss effects in the track fitting procedure become less important as the particle momentum increases, track fitting models which account for the average energy loss should obtain improved chi-square values for all tracks and provide more accurate reconstruction of all particle momentum vectors at the entrance to the TPC. For low momentum ($\sim$ 150 - 200 MeV/c) pions the effect is small, for low momentum kaons it is significant, and for low momentum protons the effect can be substantial. Furthermore, for low momentum protons in the TPC the variation in dE/dx along the trajectory due to the reduction in particle momentum also becomes important. In this Starnote a constant dE/dx is assumed, or more accurately, constant *momentum loss* $(E/p)(dE/dx)$, all along the particle's trajectory. A model with varying momentum loss will be addressed in a later Starnote.

Trajectory Model

The model was developed specifically for constant, uniform magnetic fields oriented along the $+z$ axis (as in the STAR TPC). As a charged particle (for now assume a negative charge resulting in c.c.w. motion) moves through an incremental path length Δs the increment in phase angle of rotation about the center of track curvature, $\Delta\xi$, is given by

$$\Delta\xi = \frac{\Delta s\ \cos\theta}{R(\xi)} \tag{1}$$

where θ is the pitch angle and $R(\xi)$ is the track radius of curvature at phase angle parameter ξ. The incremental change (reduction) in momentum is given by

$$\Delta p = \frac{E}{p}\frac{dE}{dx}\Delta s \tag{2}$$

where E and p are the total energy and momentum, respectively, and dE/dx (here taken to be negative) is the mean energy loss per unit length for this charged particle in the TPC gas at this total momentum. The incremental change (reduction) in track radius of curvature is,

$$\Delta R = \frac{\Delta p\ \cos\theta}{\mid B_z q \mid}\ . \tag{3}$$

If the units for R, p, B_z and q are cm, GeV/c, T, and e, respectively, then

$$\Delta R = \frac{\Delta p\ \cos\theta}{0.002998\ \mid B_z q \mid}\ . \tag{4}$$

Using Eqs. (2) and (3) we define

$$\Delta R \equiv K_0 \Delta s \tag{5}$$

where

$$K_0 = \frac{E}{p}\frac{dE}{dx}\frac{\cos\theta}{0.002998\ \mid B_z q \mid} \tag{6}$$

which is negative definite. From Eq. (5) we have

$$\Delta R = \frac{K_0}{\cos\theta} R(\xi)\Delta\xi \tag{7}$$

or

$$\frac{dR}{d\xi} = \frac{K_0}{\cos\theta} R(\xi) \tag{8}$$

the solution of which is

$$R(\xi) = R_0 \mathrm{exp}\,[K_0(\xi - \xi_0)/\mathrm{cos}\theta] \tag{9}$$

where R_0 and ξ_0 are the radius of curvature and phase angle at the initial point on the track.

After moving through Δs the track radius changes by $K_0 \Delta s$. Only *momentum loss*, not multiple Coulomb scattering (MCS), is considered here. The momentum direction is not allowed to randomly change due to Coulomb scattering. This model therefore describes the average trajectory for an ensemble of particles at a given initial momentum. With these assumptions the center of the track radius of curvature changes according to

$$\begin{aligned} \Delta x_c &= -\Delta R \mathrm{cos}\xi = -K_0 R(\xi)\Delta\xi \mathrm{cos}\xi/\mathrm{cos}\theta && (10) \\ \Delta y_c &= -\Delta R \mathrm{sin}\xi = -K_0 R(\xi)\Delta\xi \mathrm{sin}\xi/\mathrm{cos}\theta && (11) \end{aligned}$$

or

$$\begin{aligned} \frac{dx_c}{d\xi} &= -\frac{K_0}{\mathrm{cos}\theta} R(\xi)\mathrm{cos}\xi && (12) \\ \frac{dy_c}{d\xi} &= -\frac{K_0}{\mathrm{cos}\theta} R(\xi)\mathrm{sin}\xi\ . && (13) \end{aligned}$$

Substituting for $R(\xi)$ and integrating yields,

$$\begin{aligned} x_c(\xi) &= -\frac{K_0'^2 R_0}{K_0'^2+1}\mathrm{exp}\,[K_0'(\xi-\xi_0)]\,(\mathrm{cos}\xi + \frac{1}{K_0'}\mathrm{sin}\xi) \\ &\quad + \frac{K_0'^2 R_0}{K_0'^2+1}(\mathrm{cos}\xi_0 + \frac{1}{K_0'}\mathrm{sin}\xi_0) + x_{c,0} && (14) \end{aligned}$$

and

$$\begin{aligned} y_c(\xi) &= \frac{K_0' R_0}{K_0'^2+1}\mathrm{exp}\,[K_0'(\xi-\xi_0)]\,(\mathrm{cos}\xi - K_0'\mathrm{sin}\xi) \\ &\quad - \frac{K_0' R_0}{K_0'^2+1}(\mathrm{cos}\xi_0 - K_0'\mathrm{sin}\xi_0) + y_{c,0} && (15) \end{aligned}$$

where $K_0' \equiv K_0/\mathrm{cos}\theta$ and $x_{c,0}$ and $y_{c,0}$ are the coordinates of the initial center of curvature.

Therefore the (x, y) coordinates of the trajectory as a function of the phase parameter ξ are given by

$$\begin{aligned} x(\xi) &= x_c(\xi) + R(\xi)\mathrm{cos}\xi && (16) \\ y(\xi) &= y_c(\xi) + R(\xi)\mathrm{sin}\xi\ . && (17) \end{aligned}$$

The integrated path length $s(\xi)$ is given by

$$\begin{aligned} s(\xi) &= \int_{\xi_0}^{\xi} ds = \frac{R_0}{\cos\theta}\int_{\xi_0}^{\xi} \exp\left[K_0'(\xi-\xi_0)\right] d\xi \\ s(\xi) &= \frac{R_0}{K_0}(\exp\left[K_0'(\xi-\xi_0)\right]-1) \end{aligned} \tag{18}$$

and inverting gives the phase angle ξ in terms of path length,

$$\xi = \frac{1}{K_0'}\ln\left(1+s(\xi)\frac{K_0}{R_0}\right)+\xi_0 \ . \tag{19}$$

Finally the z coordinate is given by

$$z(\xi) = z_0 + \sin\theta s(\xi) \ . \tag{20}$$

The coordinates of the trajectory can be generated either as functions of path length or phase angle.

The preceding equations apply for negative charged particles with B_z along the $+z$ axis. For $\pm$ charge and/or $\pm B_z$ the relevant equations can be generalized by defining a quantity ϵ_q given by

$$\epsilon_q = \frac{B_z q}{\mid B_z q \mid} \ . \tag{21}$$

With this factor included Eq. (1) becomes

$$\Delta\xi = -\epsilon_q \frac{\Delta s \ \cos\theta}{R(\xi)} \ \text{, where } \Delta s \geq 0 \ . \tag{22}$$

If we replace $K_0' = K_0/\cos\theta$ with

$$K_0' = -\epsilon_q K_0/\cos\theta \tag{23}$$

then

$$R(\xi) = R_0 \exp\left[K_0'(\xi-\xi_0)\right] \tag{24}$$

and Eqs. (14) - (20) for $x_c(\xi)$, $y_c(\xi)$, $x(\xi)$, $y(\xi)$, $s(\xi)$, ξ, and $z(\xi)$, respectively, are unchanged.

This model was applied to pion, kaon and proton trajectories in a 0.5 T magnetic field in the P10 TPC gas using the Bethe-Bloch formula for the mean dE/dx at the initial momentum. The analytic trajectories for all cases compare extremely well with those generated numerically using small incremental steps and constant K_0. For low momentum protons the additional effects caused by the increase in dE/dx along the path as the proton loses momentum are significant due to the $1/\beta^2$ dependence in the Bethe-Bloch formula ($1/\beta^3$ dependence for K_0). An analytic model incorporating this further effect is being studied.

Resulting trajectories for low momentum kaons and protons with $p_T = 0.17$ GeV/c at η=0, starting at the inner radius of the TPC (60 cm) and extending for a path length of 2.5 m are shown in Figs. 1 and 2. The assumed gas was P10 and the magnetic field was taken to be 0.5T. The dotted lines in each figure indicate zero energy loss trajectories (helices). The innermost solid curve shows the simulated trajectory with energy loss computed at each step using the local momentum value and the mean value from the Bethe-Bloch formula. The other curve is the model prediction assuming constant momentum loss. The model and simulated trajectories are indistinguishable for the kaon and are very similar for the proton. The effects for pions at similar momenta (not shown) and for any higher p_T particles are much less important.

Distance of Closest Approach

In fitting trajectories to space points one needs to know the distance of closest approach in the $x-y$ plane between the analytic function and the space points. Also needed are the path length s and the z coordinate. It is assumed that a two-step track fitting procedure will be used where $x_{c,0}$, $y_{c,0}$, R_0 and K_0' are determined by fitting in the $x-y$ plane, followed by a determination of θ by fitting in the $s-z$ plane.

Thus the position along the trajectory closest (in the $x-y$ plane) to the particular space point must be found. The usual distance minimization method results in a transcendental equation which requires a numerical or iterative solution. The following method yields accurate results (±0.001 rad) for ξ in 2 to 3 iterations.

Let the specified space point coordinates be x_{exp} and y_{exp} and the closest point of approach along the trajectory be located by $\bar{\xi}$ or $\bar{s}$. For the initial trial value for $\bar{\xi}$ use the phase angle determined by $(x_{c,0}, y_{c,0})$ and (x_{exp}, y_{exp}). With this value of $\bar{\xi}$ determine new values of (x_c, y_c) from Eqs. (14) and (15). Using this new center and (x_{exp}, y_{exp}) find a new trial value for $\bar{\xi}$. Repeat this process until $\bar{\xi}$ is sufficiently stable and obtain the final path length, coordinates and distance of closest approach. A Fortran subroutine, CLOSE_POINT (see appendix), was written which computes the point of closest approach given the parameters of the momentum loss trajectory, based on this iterative method.

Summary

An analytic model was derived and tested which describes the average trajectory of charged particles moving in constant magnetic fields with constant momentum loss per unit path length. The model includes one additional parameter compared to the usual helix parametrization. A computer code which calculates the minimum distance between fixed space points and the trajectory model was provided. The latter is required in order to fit the model trajectory to reconstructed space points in the TPC.

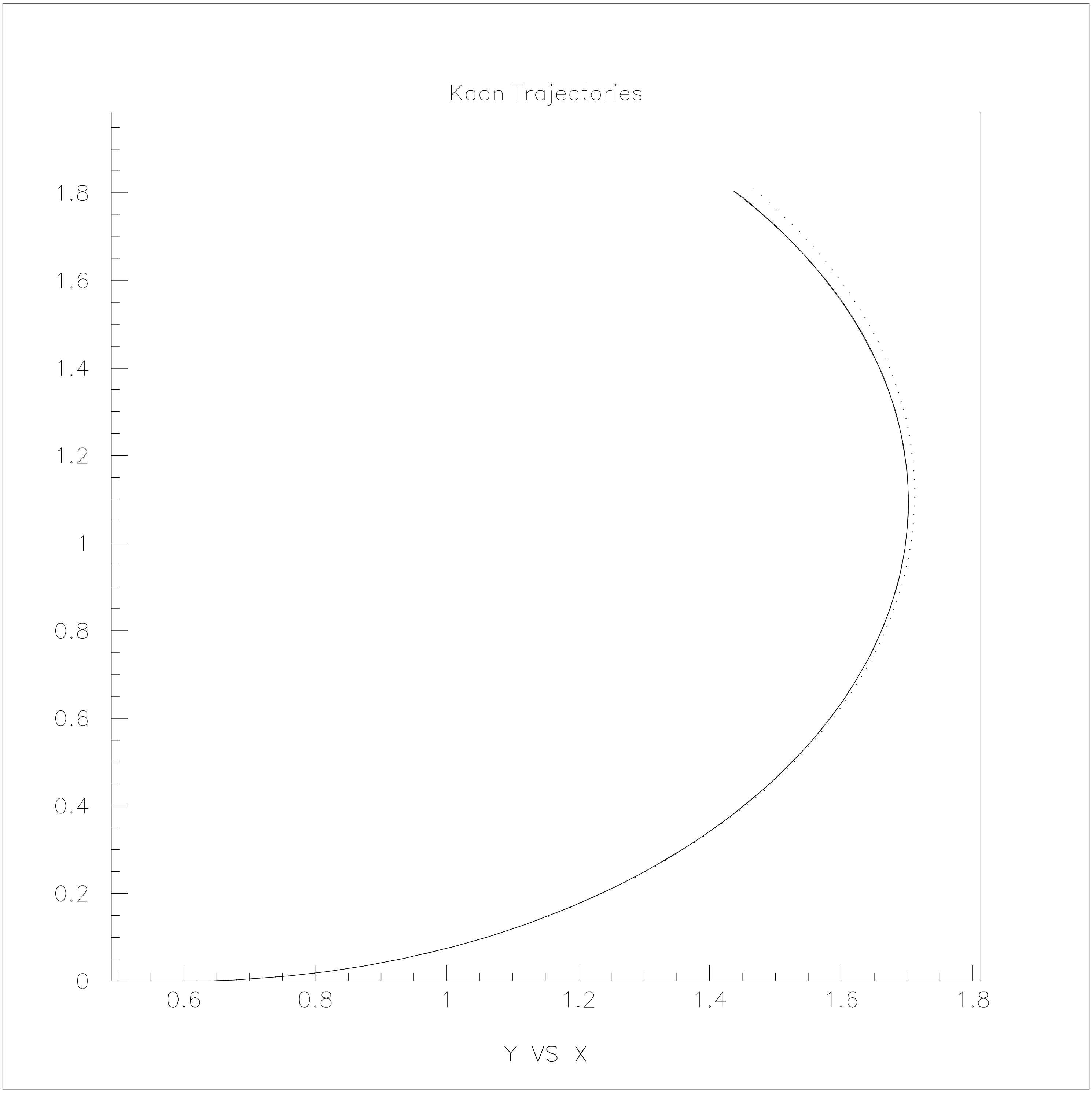

Figure 1: Trajectories for $\eta = 0$, p_T=0.17 GeV/c kaons in the STAR TPC (P10 gas, $B_z = 0.5$T, no MCS) with (solid line) and without (dotted line) momentum loss.

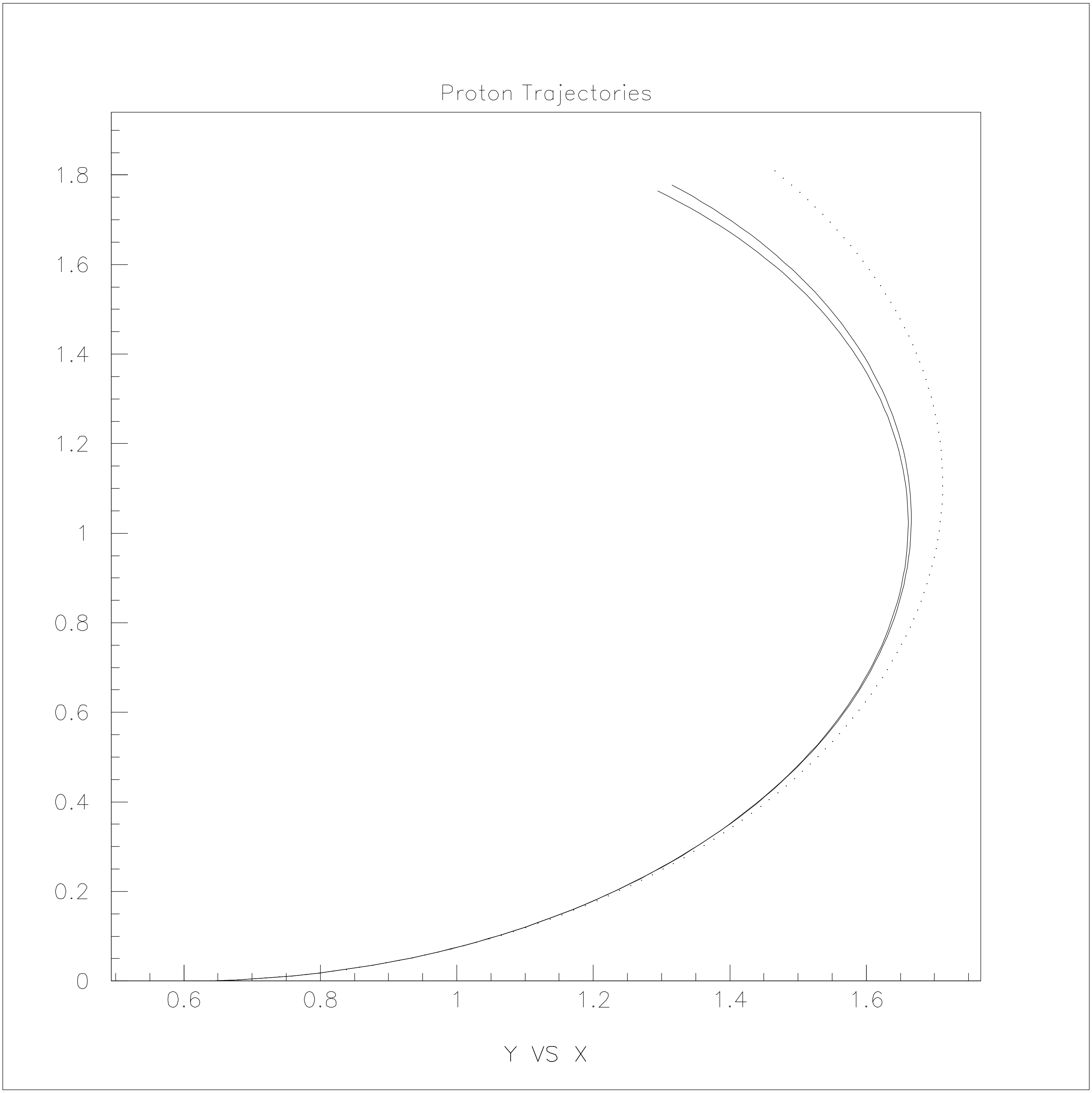

Figure 2: Trajectories for $\eta = 0$, p_T=0.17 GeV/c protons in the STAR TPC (P10 gas, $B_z = 0.5$T, no MCS) with (solid lines) and without (dotted line) momentum loss.

Appendix: Listing of Fortran code for momentum loss trajectory model.

```
      subroutine close_point(rho,phi0,z0,invpt,psi,tanl,epsq,K0,bmag,
     1     x0,y0,z_0,ci_bar,s_bar,st_bar,x_bar,y_bar,z_bar,d,
     2     converge,max_iter)

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C                 Analytic Trajectory Model
C
C     For charged particles moving in a constant, uniform magnetic field
C     along the z-axis with constant energy loss (dE/dx) or more precisely
C     with constant momentum loss [(E/p)(dE/dx)].
C
C   This subroutine finds the point on the energy loss trajectory
C   that is closest (in the x,y plane) to the given input space point.
C   It also returns the distance of closest approach in the x-y plane.
C   It is assumed that the requested space point is "forward" along the track
C   from the initial point given by rho,phi0, and z0.
C
C   Input:   Quantities supplied by the calling program
C
C                 rho = cylindrical coordinate radial distance to initial
C                       point on track (cm)
C                 phi0 = azimuthal angle to initial point on track (deg)
C                 z0  = z coordinate of initial point on track (cm)
C                 invpt = 1.0/pt at initial point on track (GeV/c)**(-1)
C                 psi = azimuthal angle of momentum vector projection on x-y
C                       plane (deg)
C                 tanl = tan(pitch angle) where tanl>0 means the particle moves
C                        toward increasing values of z as time increases (for
C                        either + or - charge).
C                 epsq = Bz*q/abs(Bz*q) where Bz is the z component of the
C                        magnetic field and q is the charge
C                 K0  =  energy loss factor defined as (change in radius of
C                        curvature of track)/(increment in path length thru
C                        material). Note K0 < 0 .
C                 bmag = z component of magnetic field in T
C
C                 x0,y0,z_0 are the coordinates of the requested space point (cm)
C
C                 converge = minimum acceptable error in the iterative method
```

```
C                             used to evaluate the phase angle at the closest
C                             point (radians)
C                 max_iter = maximum number of iterations allowed to find closest
C                             point of approach on track.
C
C         Note:   Typically only 2 to 3 iterations are required to get the phase
C                 angle accurate to 0.001 radians which should be sufficient.
C
C   Output:   Quantities generated by the subroutine
C                 ci_bar = phase angle (rad) of the closest point of approach
C                           on the trajectory.  Refer to notes.
C                 s_bar  = path length (cm) from initial point to the closest
C                           point of approach along actual trajectory.
C                 st_bar = projected (x-y plane) path length (cm) from initial point
C                           to the closest point of approach.
C                 x_bar,y_bar,z_bar are the coordinates of the closest point of
C                           approach on the trajectory.
C                 d      = sqrt((x_bar - x0)**2 + (y_bar - y0)**2)
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      implicit none

      real rho,phi0,z0,invpt,psi,tanl,epsq,K0,bmag
      real x0,y0,z_0,ci_bar,s_bar,st_bar,x_bar,y_bar,z_bar,d
      real converge
      real r0,x0_cntr,y0_cntr,pi,rad,ci0,cos_theta,K0p
      real radius,facxc,facyc,ci_trial(2),expK0
      real x_center,y_center,diff
      integer max_iter, icount
      real fix_mod2pi

      pi = 3.141592654
      rad = 180.0/pi

C   convert track parameters

      r0 = 1.0/(invpt*abs(bmag)*0.002998)
      x0_cntr = rho*cos(phi0/rad) + r0*cos(psi/rad - epsq*pi/2.0)
      y0_cntr = rho*sin(phi0/rad) + r0*sin(psi/rad - epsq*pi/2.0)
      ci0 = atan2(rho*sin(phi0/rad) - y0_cntr,rho*cos(phi0/rad)
     &      - x0_cntr)
```

```
      cos_theta = 1.0/sqrt(1.0 + tanl*tanl)
      K0p = -epsq*K0/cos_theta
      facxc = K0p*K0p*r0/(K0p*K0p + 1.0)
      facyc = K0p*r0/(K0p*K0p + 1.0)
      icount = 0
      ci_trial(1) = atan2(y0 - y0_cntr, x0 - x0_cntr)
      ci_trial(1) = fix_mod2pi(ci_trial(1),ci0,epsq,pi)
99    icount = icount + 1
      expK0 = exp(K0p*(ci_trial(1) - ci0))
      radius = r0*expK0
      x_center = -facxc*expK0*(cos(ci_trial(1)) + sin(ci_trial(1))/
     &         K0p) + x0_cntr
     &           +facxc*(cos(ci0) + sin(ci0)/K0p)
      y_center = facyc*expK0*(cos(ci_trial(1))  - K0p*sin(ci_trial(1)))
     &           + y0_cntr
     &           -facyc*(cos(ci0) - K0p*sin(ci0))
      ci_trial(2) = atan2(y0 - y_center, x0 - x_center)
      ci_trial(2) = fix_mod2pi(ci_trial(2),ci0,epsq,pi)
      diff = abs(ci_trial(2) - ci_trial(1))

      if(icount .le. max_iter) then
           if(diff .gt. converge) then
                ci_trial(1) = ci_trial(2)
                go to 99
           else
                ci_bar = ci_trial(2)
                x_bar = x_center + radius*cos(ci_bar)
                y_bar = y_center + radius*sin(ci_bar)
                d = sqrt((x_bar - x0)**2 + (y_bar - y0)**2)
                expK0 = exp(K0p*(ci_bar - ci0))
                s_bar = (r0/K0)*(expK0 - 1.0)
                st_bar = s_bar*cos_theta
                z_bar = z0 + tanl*st_bar
C     write(6,55) icount
C55   format(5x,'icount = ',i10)

                return
           end if
      else
C          write(6,100) converge, max_iter
C100       format(5x,'Iteration failed to converge to ',e12.5,
C    &            ' after ',i10,' iterations')
```

```
end if

end

function fix_mod2pi(ci,ci0,epsq,pi)
implicit none
real ci, ci0, epsq, pi, fix_mod2pi
fix_mod2pi = ci
if(epsq .lt. 0.0) then
     if((ci - ci0) .lt. 0.0) fix_mod2pi = ci + 2.0*pi
else
     if((ci - ci0) .gt. 0.0) fix_mod2pi = ci - 2.0*pi
end if
return
end
```