Geane Track Fitting

Nicholas Kinnaird

April 13, 2017

Abstract

In this document I will detail the concepts and mathematics of GEANE track fitting, as well as it's implementation in the gm2 simulation framework. I will also detail the pecularities and intricacies of the code. This is done both for documentation purposes as well as a precursor to sections of my future thesis.

1 Introduction

Neccessity for tracking? Short intro about hardware? - don't go into electronics

The Muon g-2 Experiment at Fermilab will use tracking detectors in order to measure positron trajectories for the purpose of determining the beam distribution and characteristics, both for the final result and for general experiment operation. (Sort of a diagnostics tool as well as a secondary detector.) There are 3 tracker stations location at the 0, 12, and 18th sections of the ring, counting clockwise from the top most point of the ring where the inflector resides. Each station consists of 8 tracking modules arranged in a staircase pattern that follows the curvature of the ring. The upstream and downstream trackers reside just after and before vacuum chamber wall respectively. Each tracker module consists of 4 layers of straws with a stereo angle of 7.5 degrees, the first two "U" layers oriented with the tops of the straws at a greater radial position, and the second two "V" layers oriented with the bottoms of the straws at a greater radial position.

Link pictures to trackers here for posterity - pictures of trackers in ring, individual module, etc.

1.1 Other physics

Physics notes here/elsewhere? Due to the CBO of the beam, 1 tracker accounts for 90% of the physics, and a second tracker accounts for much of the remaining. (Find/learn sources for this to explain a bit better..) A large percentage of tracks don't hit much of the trackers as the curve in tightly close to the calorimeter. ...

2 In the gm2 Framework

2.1 Branches

The GEANE track fitting code is in the main develop branches, except possibly some updates within the feature/trackDevelop branch of gm2tracker.

2.2 fcl Files, Geometry, and Material

Event generation before fitting is done using the mdc fcl files in gm2ringsim using the main simulation, where one should also include the tracker dummy plane geometry for truth comparison. At time of writing there are some modified mdc#-geane fcl files which can readily be used. Only the necessary straw geometry and associated mother volumes, as well as the parallel world dummy planes are necessary to fit tracks, though including other geometry will of course change particle behaviour. All 3 main trackers are included in their original positions. There is a fcl parameter useSD to turn on and off the straw sensitive detectors so that in the reconstruction phase hits are not regenerated.

Reconstruction is performed using RunGeane.fcl in gm2tracker on the output from the above. Parameter useSD for the straws should be turned off. The fcl file loads all 3 trackers necessary for symbol and name definitions, but such that tracker 0 is rotated such that the tracking planes are parallel to the global geant X axis (using the rotateArcTracker fcl parameter for the ArcService). This is done to avoid the issue of error propagation instability close to the Z axis, as detailed in DocDB 4567 while at the same time still observing the correct azimuthally symmetric 2D field for the tracks. Track hits are rotated from their separate trackers to this one tracker frame with the geaneWorld0, geaneWorld12, and geaneWorld18 transforms defined at the bottom of StrawTrackerService.cc (or the cadmesh version). In the future, if fields or geometries are not identical between trackers, the details here will have to be improved.

Fcl parameters exist for the StrawTrackerService and StrawsService in order to turn material on or off at will, "materialTracker" and "material" respectively. The rest of the geometry has to be manually changed and rebuilt in order to remove material. These include VacuumChamberCadMesh and the World, as well as the "buildSupportPost" option in strawtracker.fcl and strawtracker_standAlone-staggered.fcl, "buildTrolly" in vac.fcl (where the associated material is hardcoded in), and "trolleySupportMaterial" also in vac.fcl. Make sure to have the same material settings for the reconstruction as how the events were generated.. This is purely for debugging the tracking.

The reconstruction code links to geaneFitParams.fcl in gm2tracker which has many variables for changing momentum spreads, straw measurement errors, random seeds, toggling truth info, etc. Things get more complicated here so ask me before trying to play with this stuff.

2.3 GEANE TrackFitting

The GEANE track fitting routines consists of a couple of main files within gm2tracker. Track candidates from upstream in the reconstruction chain are the input to this code. A single GEANEArtRecord is produced for each event or track, and is passed from one method to another, constantly being updated as the track fitting iterates along. See below for a quick summary of the method flow, and a table for the data objects within GEANEArtRecord.

1. **GeaneReco_module.cc**
   This file consists of the main code which sets up the GEANE track fitting, pulls in upstream data products, interfaces with both Geant4 and the gm2 art framework, and outputs TrackStateArtRecords and TrackArtRecords for use downstream. It's methods consists of:

   - produce
     General overriden produce method. Calls main track fitting code on an event by event basis. Produces data products for downstream use.
   - InitializeGEANE
     Geant4 GEANE initialization method called by the constructor, once per run.
   - trackFitting
     Sets up variables and objects for track fitting after reading in upstream data products (track candidates and dummy planes). Contains main track fitting iteration loops. Also contains complex L/R sequence checking routines which can be ignored for now (and probably need to be changed at some point).
   - modifyMeasuredParams
     Short method for L/R sequence checking stuff, can be ignored for now. Just modifies measured parameters based on particular sequence
   - errorPropagation
     This method tracks particles through the detector using Geant4 error propagation routines with the correct geometry and field. It builds transport matrices, error matrices, and predicted parameters which are the objects used for fitting the track. It tracks on a plane by plane and step by step basis. These routines can be used to track particles forwards and backwards, where the forwards tracking is used in the code. Changing to backwards tracking would be very non-trivial. Add more detail here later..

- angleCorrection
  Method to iteratively correct measured parameters from a radial DCA value to a U or V value based on the momentum of the track and approximating a constant field within the straw. Also corrects the errors using a simple straight line approximation which is good enough.

2. **GeaneFitter.cc**
   This file consists of the main track fitting chi2 algorithm. It multiplies measured parameters, predicted parameters, error matrices, and transport matrices together to produce a chi2 for the track and an improvement to the starting paramters. It also reads in the measured hits errors in order to properly fit the track. There is a fcl parameter matrixDebug which can be used to turn on or off the many large matrix cout debugging statements.

   - TrackCorrelation
     The main matrix multiplication routines.
   - preSequenceChecking
     A method in order to create a hybrid wire / U or V error matrix for L/R sequence checking. Only called once per event.
   - sequenceChecking
     The main L/R sequence checking method for an individual event. This method is called many thousands of times as each U or V sequence is checked. Ignore for now.
   - convertTo
     Some methods for converting Eigen 5 vectors from GeV cm to MeV mm and vice versa.

3. **GeanePlots_module.cc**
   The main analyzer/plotting module which runs on the output GEANEArtRecords produced upstream. This creates many plots including chi2 distributions, p value distributions, number of iterations, track parameter, residuals, pulls, etc. Truth information for these plots is currently necessary as written, but can be take out in the future for real data. There are some fcl paramters to make cuts on different parameters. There are also some other plotter modules which are similar but much reduced in scope.

4. **GEANESingleEventViewer_module.cc**

5. **GeaneFailedPlots_module.cc**

6. **GeanePoorEvent_module.cc**

7. **refit geane stuff?**

Quick summary of the track reconstruction method flow: The main file StrawTrackLeftRightGEANE_module is called and sets up both the Geant4 world and the GEANE routines. Produce is called per event, a GEANEArtRecord is created, and upstream track candidates are pulled in and the relevant parameters are converted to necessary track fitting objects within the method trackFitting. The methods errorPropagation, angleCorrection, and TrackCorrelation (in GEANEFitter) are called iteratively within a loop until track fitting succeeds or fails. The relevant parameters of the updated GEANEArtRecord are then converted to TrackArtRecords and TrackStateArtRecords for downstream use, and the analyzer/plotter modules are called for output checking.

Note: Ignore copious LR ambiguity solving code for now. When last checked it was working pretty well but not perfectly, and it needs to be studied more. Many changes have been made to other code which will probably break the interfaces to the LR code as well. Also all of this LR stuff might be pulled out and placed into other tracking code modules before and after the GEANE code, but that is unclear.

Table 1: GeaneArtRecord.hh active variables. This is subject to change. GeaneArtRecord contains vectors of variables on planes, as well as larger objects containing information about the whole track. improve comments here and in the code - don't forget to update this if/when I change things in the code

art::Ptr < gm2strawtracker::TrackCandidateArtRecord > candidate *One track corresponding to one candidate corresponding to one GEANEArtRecord for the whole track.*
std::vector< art::Ptr< gm2truth::GhostDetectorArtRecord > > dummyPlaneHits *Associated dummy plane hits on planes aligned with straw wires, vector consists of hit dummy planes corresponding to hit wire planes (if a straw plane was skipped but that dummy plane was hit, it is not included in this vector. Vector has size N = num hits in straws that form the track.)*
int failureMode *Different failure modes for failed track reconstruction, 0 means it passed.*
double chi2 *Chi2 for whole track.*
std::vector<double> chi2Iterations *Chi2s for whole track for different iterations.*
std::vector<double> chi2Planes *Individual chi2s on each plane which add up to total chi2, vector consists of hit planes with size N.*
int numIterations *Number of iterations to converge.*
unsigned int dof *DoF of track = number of hit planes - 5 track parameters.*
double chi2DoF *chi2/dof*
double pValue *Fit pValue for whole track.*
double energyDiff *energy loss between first and last hit in track - from truth, for material characterizing*
std::vector<G4double> startingTrackParameters *Starting parameters for track: size 6, 3 position then 3 momentum, x y z px py pz, best starting parameters updated after each iteration, starting parameters x position defined before first hit.*
std::vector<G4double> startingTrackGuessOffsets *Size 10, x y z px py pz p 1/p pu/px pv/px offsets in different starting track parameters for plotting purposes.*
int trackNumPlanesHit *Total number of planes hit.*
int trackFirstPlaneHit
int trackLastPlaneHit
std::vector<int> trackPlanesHitList *List of hit planes, with missed planes excluded from the vector. Ex. 1 2 4 5 8 9*
Sequence information excluded from this table for now.
std::vector<std::vector<G4double> > wireUVPositions *Wire center U and V postions, first vector is track param vector size 5 (0 1 2 unfilled, 3 is U, 4 is V), second vector is planenumber from 0 - 32 (formatted this way to align with other similar vectors - can probably be reduced.)*
std::vector<G4double> measuredDCAs *Vector of measured DCAs for hit planes, with size 33. Mainly to hold on to smearing values for now.*
std::vector<G4double> UVerrors *Vector of UV measurement errors for planes 0-32. Built in order to accomadate varying errors in the future.*
std::vector<G4double> planeXPositions

Vector of X postions of hit wire planes with size 33 (0 - 32), 0 plane being in front of the first module that was hit.
std::vector<std::vector<G4double> > geaneMeasuredParameters *Measured GEANE parameters, first vector is param num 0 - 4, second vector is plane number 0 - 32, units are MeV mm. 1/P, Pu/Pz, Pv/Pz, U, V - only U or V is filled at the start of the GEANE fitting module.*
std::vector<std::vector<G4double> > geanePredictedParameters *Predicted GEANE parameters, first vector is param num 0 - 4, second vector is plane number 0 - 32, units are MeV mm. 1/P, Py/Px, Pz/Px, Y, Z - all params filled in tracing stage of GEANE fitting module - coord system has to be orthogonal - converted to UV locally in the fitting module.*
std::vector<Eigen::MatrixXd> geaneTransportMatrices *Units of GeV cm - transport matrices between planes tracked to in GEANE fitting module, 5x5 objects. Vector has size 33.*
std::vector<Eigen::MatrixXd> geaneErrorMatrices *Error matrices on tracked to planes, units GeV cm, size 33.*
std::vector<Eigen::VectorXd> extendedParamPredictedInUVEigen *Predicted parameters in UV space as an eigen object (converted from geanePredictedParameters above) for calculation convenience and some LR information storage. Order of vectors is switched here, first is planenum, second is paramnum, units are MeV mm.*
Objects below here are full track objects with larger sizes, mostly held on to for fast sequence checking, with some used in other parts of the code. Units GeV cm.
std::vector<Eigen::MatrixXd> extendedTransportMatrixBegToEnd *Accumulated/combined transport matrices from starting plane to all following planes.*
Eigen::MatrixXd extendedCombinedTransportMatricesTranspose *Transpose of larger eigen object composed of above begtoend transport matrices.*
Eigen::MatrixXd extendedCovarianceTotal *5x5 total covariance matrix for track, except* ***INVERTED****.*
Eigen::MatrixXd extendedReducedMatrix *Total error correlation matrix, reduced to size NxN (N = num planes hit).*
Eigen::MatrixXd extendedReducedMatrixInverseLargeErrors *Inverse of above but with larger errors from initial wire fitting, for LR checking.*
Eigen::MatrixXd extendedModifiedReducedMatrix *Reduced matrix from above that's going to be modified into a hybrid error matrix separately for U and V fits but that needs to be held onto for all sequences.*

2.4 Code details

Building and multipilying transport and SC2SD matrices Modifying measured parameters based on momentum angle/field and previous predicted Converting things to eigen objects

JacobianToUV stuff - or combine with other coordinate transform stuff Matrix accumulation/reduction Notes on loop indices, start points and such

2.5 Code peculiarities / notes

Where to structure this in the file?

Notes on coordinate systems I use - UV pointing outward vs XYZ, XUV x being forward, UVW in geante src, maybe IJK/TUV in one paper-prob not, GeaneTrackerWorld[0,12,18], rotateArcTracker (bit more specific), magnetic field access coordinates?, JacobianToUV

accessing the field in a specific way

3 gm2Geane package in artg4 - eloss, brems, ionizations, msc

See many docdbs by Nick Kinnaird on geane updates for more info if wanted. Summary of stuff here.

There is a folder under artg4/gm2Geane, where slightly (but importantly) modified geant/geane code that the geane track fitting uses is located. Most importantly is that the reconstruction was taking too much energy away from the particle during propagation due to bremsstrahlung, so that has been removed. (Thanks James for help with this.) (Probably explain flow of file - extrapolator tables etc.).

Besides those associated files, all files from the geant folder "error_propagation" have been copied into this directory as well, with the only non-name changes lying in gm2GeaneFreeTrajState.cc. (Many files had to be copied due to naming/linking - some could probably have been omitted but I decided to copy the whole folder.) Ionization and multiple scattering errors have been modified according to Lavezzi thesis (explain more) - with no improvement in results unfortunately. (Probably turn defaults back to original code (still gm2Geane) with the option to turn things on somehow if desired - at least leave them in for future tuning.)

Should I go more in depth about the error_propagation/gm2Geane source code? Probably - at least for certain main files.

Add many plots and pictures of things - pulls, single events, tracks, etc etc.

Figure 1: caption - landau tail, from ionizations, brems, blah blah

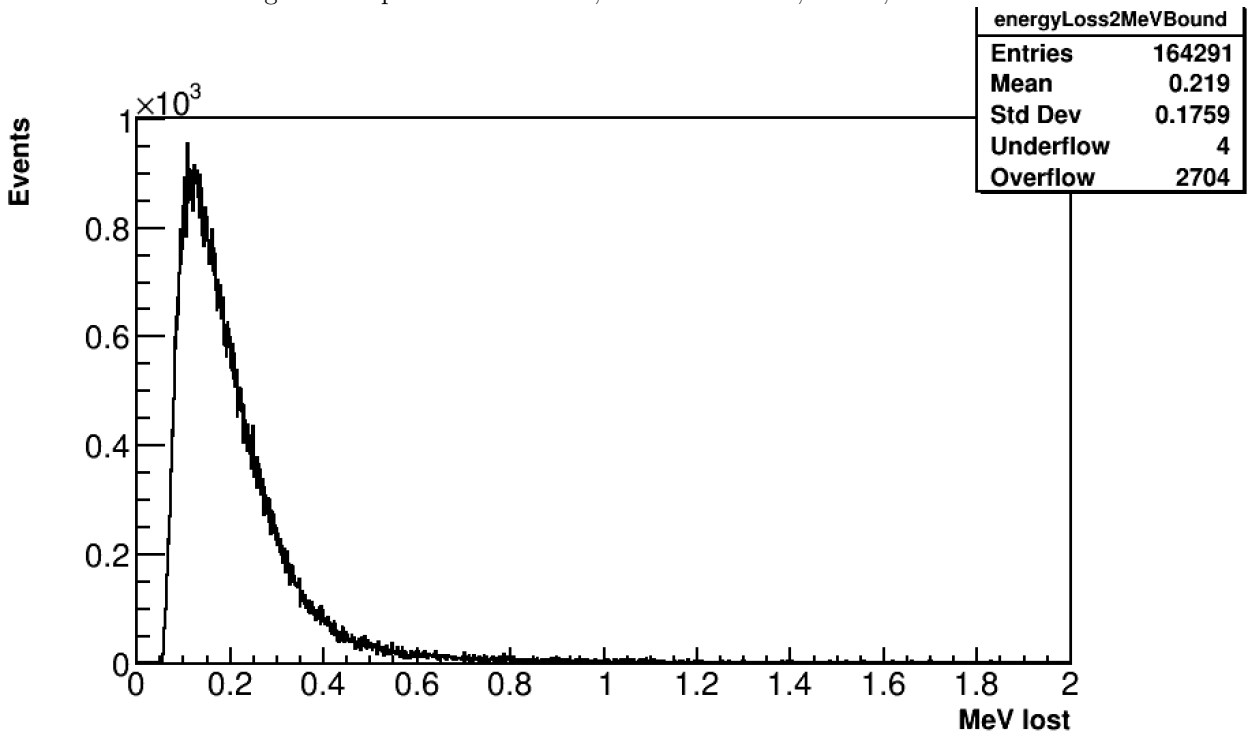
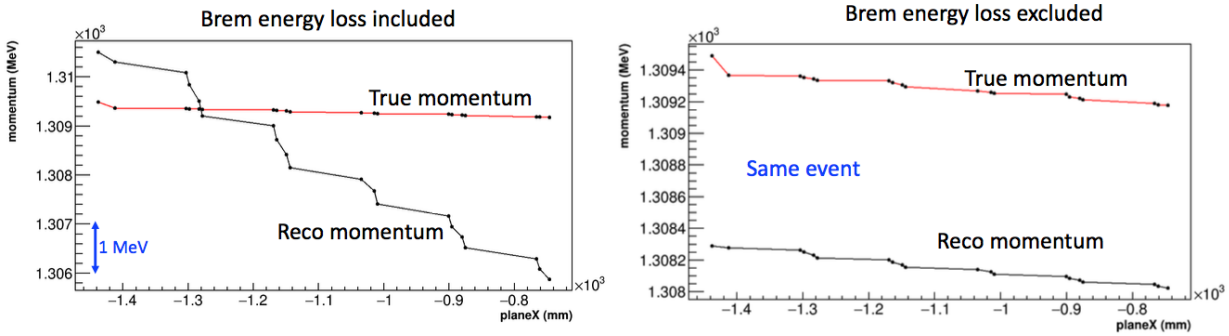


Figure 2: caption - brems were adding too much energy back in - fixed event, blah blah, notice scale change



4 notes and other

starting error matrix
linking equations to code variables
initial errors into tracing/error propagation?

5 Mathematics - Appendix maybe? - all of these should be cited but maybe organized better in someway

I need to summarize things locally a la geane manual - except with my own changes and explanations.
GEANE Manual
Trajectory fit in presence of dense materials
Derivation of Jacobians for the propagation of covariance matrices of track parameters in homogeneous magnetic fields
Lavezzi thesis - The fit of nuclear tracks in high precision spectroscopy experiments
Energy loss in thin layers in GEANT

6 Plots

It'd probably be good to make one big set of plots to toss into the back for people to look at/reference. All sorts of pulls, parameter plots, residuals, etc. Sort of like Renee's verification pdf or whatever.