G4beamline Validation

Tom Roberts
Muons, Inc.
tjrob@muonsinc.com

January 2011

http://g4beamline.muonsinc.com

G4Beamline Validation
Contents

1 Introduction

G4Beamline [1] is a single-particle tracking simulation program based on the Geant4 toolkit [2]. Its primary use is evaluating designs of new beamlines and charged-particle beam facilities. So it is important that it simulate real facilities as accurately as possible, and that its accuracy be quantified as well as possible. This document describes the many tests of G4beamline and Geant4 accuracy that have been performed.

As most of the physics processes implemented in G4beamline come directly from Geant4, the validation performed by the Geant4 collaboration is directly applicable. This document will merely summarize their results, and provide references. Some particularly important processes, such as transport and multiple scattering, have been tested independently, and those results are reported here.

This document does not discuss how to use G4beamline; for that, see the "G4beamline User's Guide" [1]. The input files used for specific tests of G4beamline are included in the Validation directory of the distribution, starting with release 2.8.

Except as noted, the tests described here were performed using G4beamline 2.06, which uses Geant.9.3p01. Most of the Geant4 Collaboration's tests used earlier versions.

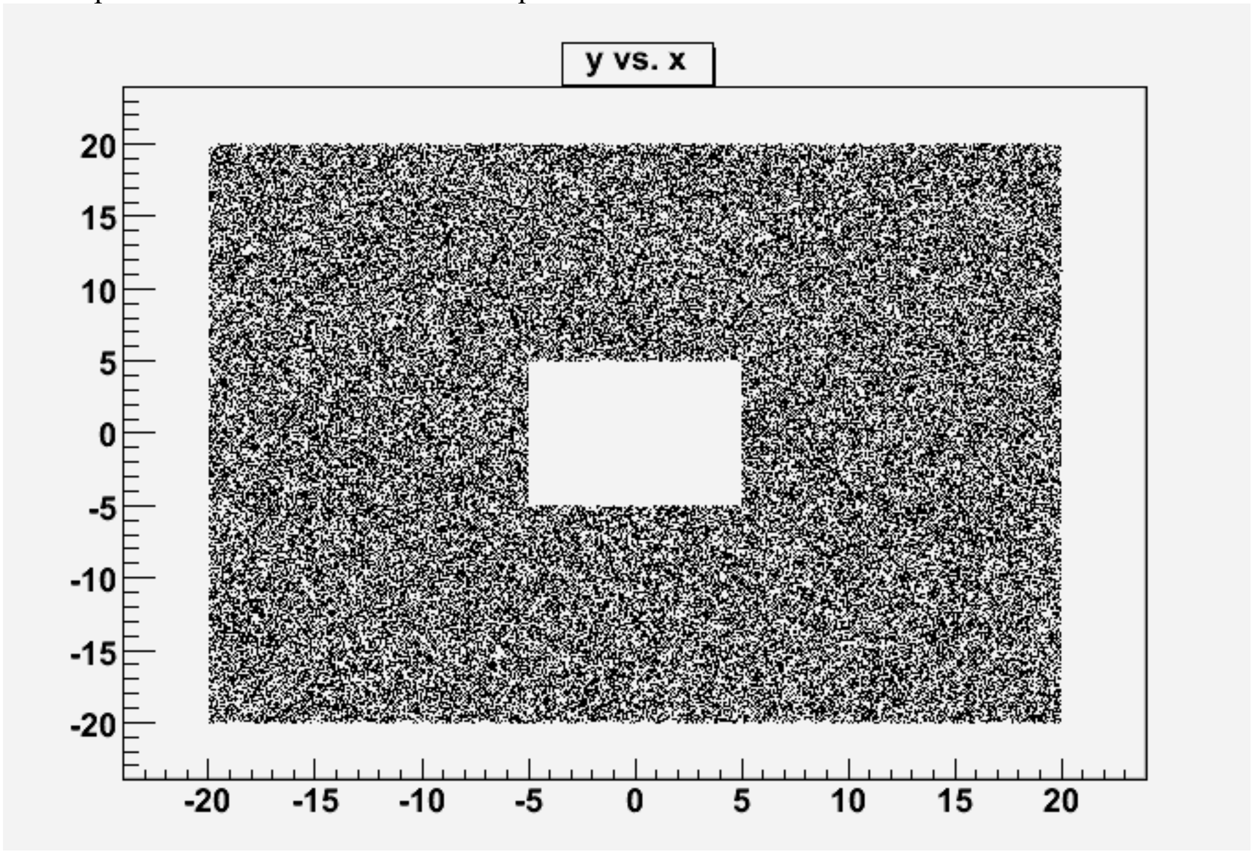
Some authors distinguish between verification and validation, using the first to mean testing that the code performs as intended, and the second to mean its physics accuracy. This document makes no such distinction – if the code is wrong the physics cannot be correct.

2 General

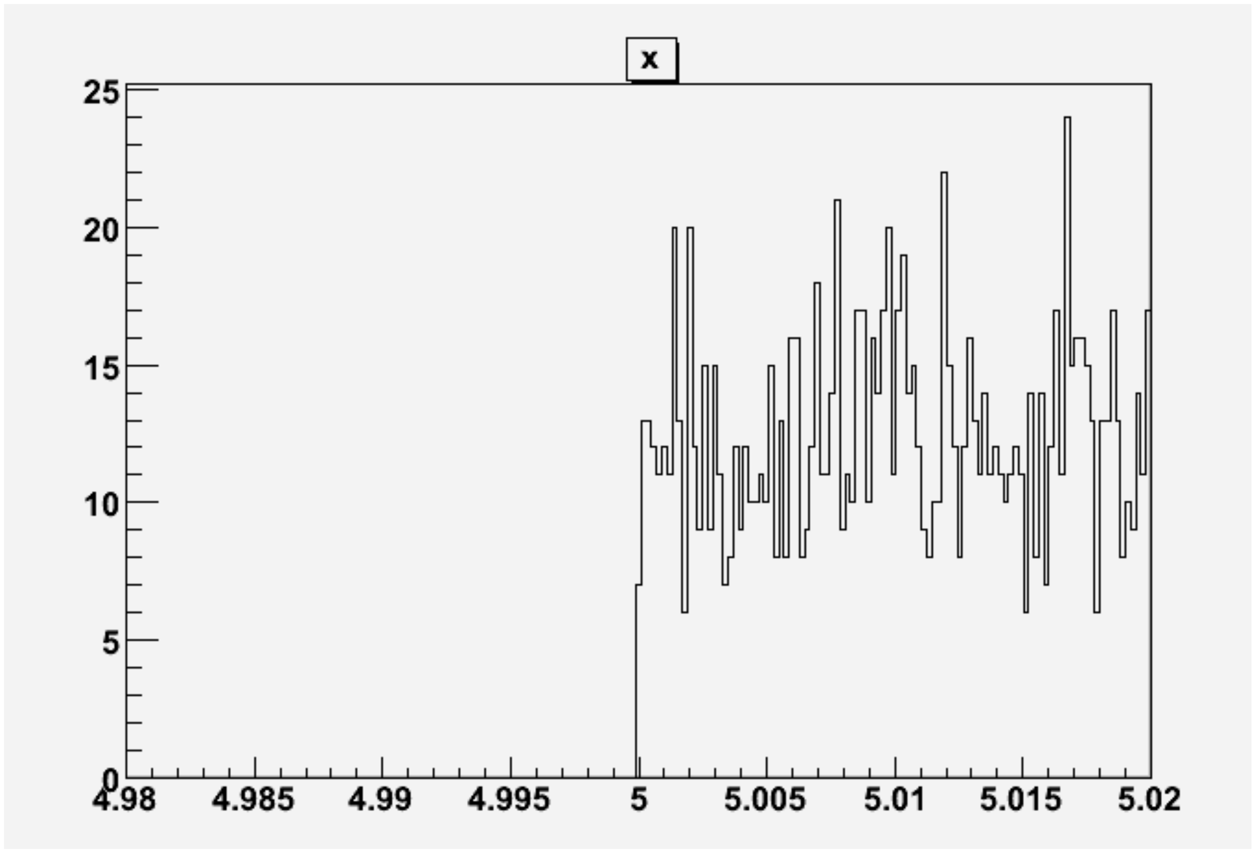
These are general tests, unrelated to specific physics processes or beamline elements.

2.1 Element Sizing and Placement

All element sizes and placements are performed with the accuracy of a double; all rotations are specified in degrees, but decimals are permitted, again with the accuracy of a double. Here is a scatter plot of the shadow of a 10-mm square box that kills all tracks that hit it:



Here is a histogram for events with -4.9 < y < 4.9 (i.e. within the vertical region of the box), showing the right edge (x) has an accuracy better than 0.2 micron:



The input file for this test is placement.g4bl:

```
*	placement.g4bl - verify object placement with beam
physics QGSP_BERT
# this is a uniform beam emanating from a square 20 mm on a side
beam gaussian sigmaX=-20 sigma=-20 nEvents=10000000
box B height=10 width=10 length=10 kill=1 color=0,0,1
place B z=100
virtualdetector Det height=1000 width=1000 length=1 color=0,1,0
place Det z=200
```

2.2 Data Input and Output

Input tracks from files in Root [3], ASCII, or BLTrackFile format come via NTuples, which have float values. Output is similarly sent via NTuples with float values; for ASCII and BLTrackFile formats the floats are printed in %.6g format. Here's a test that directly writes 0.001 mm after reading (spaces added to line up the columns):

```
Input:    0.123456 0.1234567 0.0     0.000 0.000 200.123456 0.375       -13 0 1 0 1.0000
Output:   0.123456 0.123457   0.001 0       0       200.123      0.375004  -13 0 1 0 1
(The differences are correct.)
```

The input file for this test is dataio.g4bl:

```
*	dataio.g4bl - test BLTrackFile data accuracy
```

```
physics QGSP_BERT
beam ASCII file=in.txt
virtualdetector Det radius=1000 length=1 format=ASCII file=out.txt
place Det z=0.001 front=1
```

2.3 Pseudo-Random Number Generator

G4Beamline uses the default Geant4 random number generator. It is an excellent pseudo-random number generator that essentially guarantees no repeat sequences when seeded by any integer between 0 and 900 million. At the start of each event, G4Beamline seeds the random number generator with the event number. This permits the user to submit multiple jobs in parallel with confidence, as long as no two jobs run the same event numbers. This also permits the user to re-run the same events (as long as the simulation remains unchanged). The *randomseed* command can be used to change this default behavior.

The random number generator comes from CLHEP [4], and is the default HepJamesRandom generator, described in [5]. It is documented to provide independent random streams for seeds between 0 and 900,000,000 (inclusive), which exceeds the normal range of event numbers in G4beamline. It is not feasible to perform a useful test of this generator here, see the references for details.

2.4 Regression Tests for G4beamline Releases

Before every release of G4beamline, a set of tests is run to make sure that the program basically works, and that no previously working features were broken in obvious ways. The goal is to have at least one test for each G4beamline command; that has not quite been reached, as some commands are difficult or impossible to test in an automated manner.

In most cases, the test was written at the same time as the feature being tested. Most tests have simple physical situations and quantitative checks on some value(s) that were generated by the program; in most cases, these internal checks of the test have been verified manually. That is, most of these tests not only verify consistency among releases, but also verify that for at least one instance, the basic operation of the feature is correct.

For example, test78 tests the *setdecay* command, which permits the user to change the decay parameters of particles (a capability intended for testing and background studies). test78 does the following:

1. Changes pi+ to decay with equal branching ratios into (e+ nu_e) and (mu+ nu_mu), and a lifetime of 0.1 ns.
2. Changes mu+ to decay with a lifetime of 1E20 ns (i.e. they never decay, to avoid confusing the particle counts).
3. Makes deuterons be unstable (!), decaying into (tau+ gamma) with a lifetime of 0.1 ns.
4. Changes tau+ to have a lifetime of 1E20 ns (i.e. they never decay).
5. Runs with a beam of 50 deuterons and 100 pi+, over 10 meters.
6. Checks that at the end of the simulation (z=10000 mm) there were 50±1 tau+, 50±12 mu+, and 50±12 e+ (values outside those ranges make the test fail).

The artificially short lifetimes ensure that all pi+ and all deuterons decay (!), and the ±12 values for particle counts accommodate statistical variations. This is a highly unphysical simulation, but the presence of those particle counts at the end does imply that the *setdecay* command performed as intended. Note that this just tests for basic operation, and details such as energy and momentum conservation are not checked. The *setdecay* command has several major cases internally, and test78 was written to check one instance of each – as frequently happens, one of the unit tests of the code became a regression test.

Another example is test12 testing the *multipole* command, which creates a beamline element with a multipole magnetic field. It places 5 elements with quadrupole, sextupole, octopole, decapole, and dodecapole fields. It then uses the *printfield* command to print the B_x and B_y fields in the x-z and y-z planes inside all of them, verifying they are correct to within 0.0002 tesla. The *multipole* command has five internal cases for the different multipoles, and test12 was written to check one instance of each; this is another unit test turned into a regression test. The values of the fields were checked manually when the test was written.

A third example is test68 testing the *polycone* command, which generates an element consisting of an absorber shaped as multiple cones in a line. It creates and places a polycone, and then surrounds it with several cylinders that should or should not intersect the polycone, and checks the correctness of those intersections. It also checks that beam propagates through the polycone without error. Unit testing of the polycone command relied heavily on visualization, which cannot easily be translated into automated tests.

These tests are in the directory *test* of the G4beamline distribution. As of release 2.06 the list of these tests is:

test01: simple geometry and tracking test (1 sec)
test02: BLFieldMap test (1 sec)
test03: tracking through quads and bend (1 sec)
test04: tracking mu+ through Aluminum (5 sec)
test05: multiple successive transforms (1 sec)
test06: HistoScope omitted
test07: tracking through a group of 2 solenoids (7 sec)
test08: FOR009.DAT output (2 sec)
test09: ntuple, particlefilter test (2 sec)
test10: timentuple (1 sec)
test11: trace test (1 sec)
test12: multipole test (1 sec)
test13: tracing through FieldMaps and bend (1 sec)
test14: Eloss and multiple scattering in LH2 (15 sec)
test15: fieldexpr test (8 sec)
test16: particlefilter require argument (1 sec)
test17: various kill arguments (7 sec)
test18: various object arguments on the place command (10 sec)
test19: geometrical args on place command (1 sec)

test20: beamlossntuple (7 sec)
test21: field of rotated solenoid and solenoid cache file (7 sec)
test22: argument expressions (1 sec)
test23: place OFFSET with 2 solenoids (7 sec)
test24: virtualdetector NTuple name test (1 sec)
test25: particlefilter nWait and referenceWait test (1 sec)
test26: ntuple with for009.dat (2 sec)
test27: randomseed command (4 sec)
test28: printf (1 sec)
test29: 3 nested tune-s (2 sec)
test30: tune By of genericbend (2 sec)
test31: tune maxGradient of 4 pillbox-es (2 sec)
test32: tune reference momentum (2 sec)
test33: profile command (15 sec)
test34: element naming (1 sec)
test35: tune By of idealsectorbend (2 sec)
test36: the if and the define commands (1 sec)
test37: zntuple (2 sec)
test38: Root input and output (6 sec)
test39: zntuple + beam rotation (2 sec)
test40: MICEPhysicsList (1 sec)
test41 -- LISAPhysicsList OMITTED
test42: rotated trace (1 sec)
test43: various coordinates arguments (2 sec)
test44: steppingVerbose (1 sec)
test45: psuedo random number generator seeds and sequence (1 sec)
test46: do loops and complex (multi-line) if-s (1 sec)
test47: tune By of six idealsectorbends (2 sec)
test48: reference coordinates (2 sec)
test49: multiple beam commands eventide-s (1 sec)
test50: require arguments on NTuple commands (1 sec)
test51: newparticlentuple (2 sec)
test52: tune fieldexpr (2 sec)
test53: tune fieldmap (2 sec)
test54: Zcl (2 sec)
test55: compile and run BLMinimize (1 sec)
test56: extrusion (2 sec)
test57: eventcuts file (1 sec)
test58: NIST material database, output command (1 sec)
test59: multiple beam-s and corner-s (3 sec)
test60: tracker (4 sec)
test61: linac (3 sec)
test62: 120 GeV/c beam (10 sec)
test63: zntuple tracked both directions (2 sec)

test64: torus (2 sec)
test65: pillbox B field (2 sec)
test66: tracker (4 sec)
test67: various comma-separated list arguments (3 sec)
test68: polycone (1 sec)
test69: genericbend as a parent (3 sec)
test70: tracked preservation (1 sec)
test71: basic collective tracking (1 sec)
test72: compiling user code with g4blmake (20 sec)
test73: totalenergy (2 sec)
test74: pillbox fixed timeOffset (2 sec)
test75: pillbox kill=1 (2 sec)
test76: ntuple command (1 sec)
test77: unary minus in expressions (5 sec)
test78: setdecay command (1 sec)
test79: reference noEloss and no field (2 sec)
test80: particlesource command (1 sec)
test81: veto argument to ntuple command (2 sec)
test82: fieldntuple (2 sec)
test83: multiple reference particles, noEfield and noEloss (2 sec)
test84: extended NTuple formats (2 sec)
test85: synchrotron radiation (2 sec)

3 Physics Processes from Geant4

Most physics processes in G4beamline are taken from the Geant4 toolkit [2] without change. These processes are discussed in this section. See section 4 for physics processes implemented in G4beamline, including modifications of processes from Geant4.

The primary link for "Validation and Testing" of Geant4 from the Geant4 Collaboration is: http://geant4.web.cern.ch/geant4/results/results.shtml.

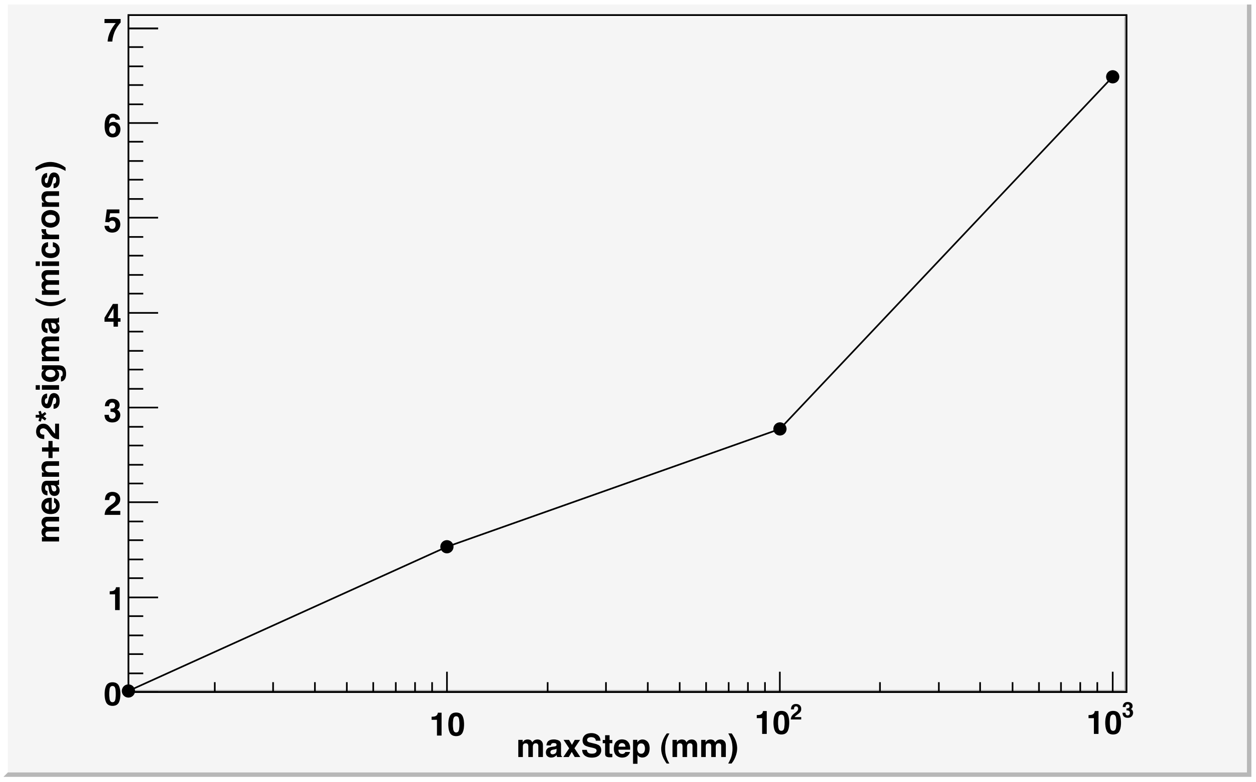
A more general link to Geant4 "Results & Publications" is: http://geant4.web.cern.ch/geant4/results/index.shtml.

The CERN "Simulation Validation Project" is also relevant: http://lcgapp.cern.ch/project/simu/validation/.

3.1 Particle Transport

Test 1

To check particle transport, one thousand 25 MeV/c electrons were tracked in the x-z plane normal to a uniform 0.01 Tesla magnetic field along y, with initial sigmaXp=0.1. These tracks have a circumference of about 52 meters. The maximum step length (maxStep) was varied from 1 mm to 1 meter, and the position in x when returning to its starting point is histogrammed for 1000 tracks. Remember that this step is the "physics step", and integrating the equations of motion in the EM field uses smaller steps controlled by the tracking parameters. In this plot, |mean|+2*sigma is plotted, providing an upper bound on the error for >95% of the tracks (fewer tracks exceed 2*sigma than for a Gaussian); the value for maxStep=1 mm cannot be read from the plot; it is 0.015 micron:



After one circle, the momentum of each track is observed to be within 0.000001 MeV/c of the initial 25.000000 MeV/c beam; all tracks have a y value within 0.01 micron of the correct value, 0.0. The transit time for each track is within 0.0001 ns of the correct value, 174.8111 ns – for these parameters the period is (MKS):

$$T = 2\,\pi\,m\,\gamma\,/\,(q\,B)$$
$$= 6.28318531 * 9.1093821\times10^{-31} * 48.934002\,/\,(1.6021765\times10^{-19} * 0.01)$$
$$= 174.8111 \text{ ns}$$

By inference, the radius of each track is correct within a few microns.

As expected, the real-time speed of tracking depends strongly on the value of maxStep:

maxStep (mm)	Tracks / sec
1	2.1
10	20.8
100	200
1000	>500

The input file for this test is *transport.g4bl*:

```
*	transport.g4bl - test G4beamline transport in a uniform B field
param -unset maxStep=100
physics QGSP_BIC
param maxStep=$maxStep histoFile=transport maxStep=$maxStep
output $histoFile.out
# beam has angular divergence, but all particles come back to (0,0,0).
```

```
beam gaussian particle=e- meanMomentum=25 sigmaXp=0.1 nEvents=1000
fieldexpr Field By=0.01 height=50000 width=50000 length=50000
place Field z=0
virtualdetector Det height=100 width=100 length=0.1 color=0,1,0
place Det z=0 front=1
particlefilter Filter height=100 width=100 length=0.1 kill=e- nWait=2
place Filter z=0.1 front=1
box W height=50000 width=50000 length=1 color=1,0,0
place W z=25000
```

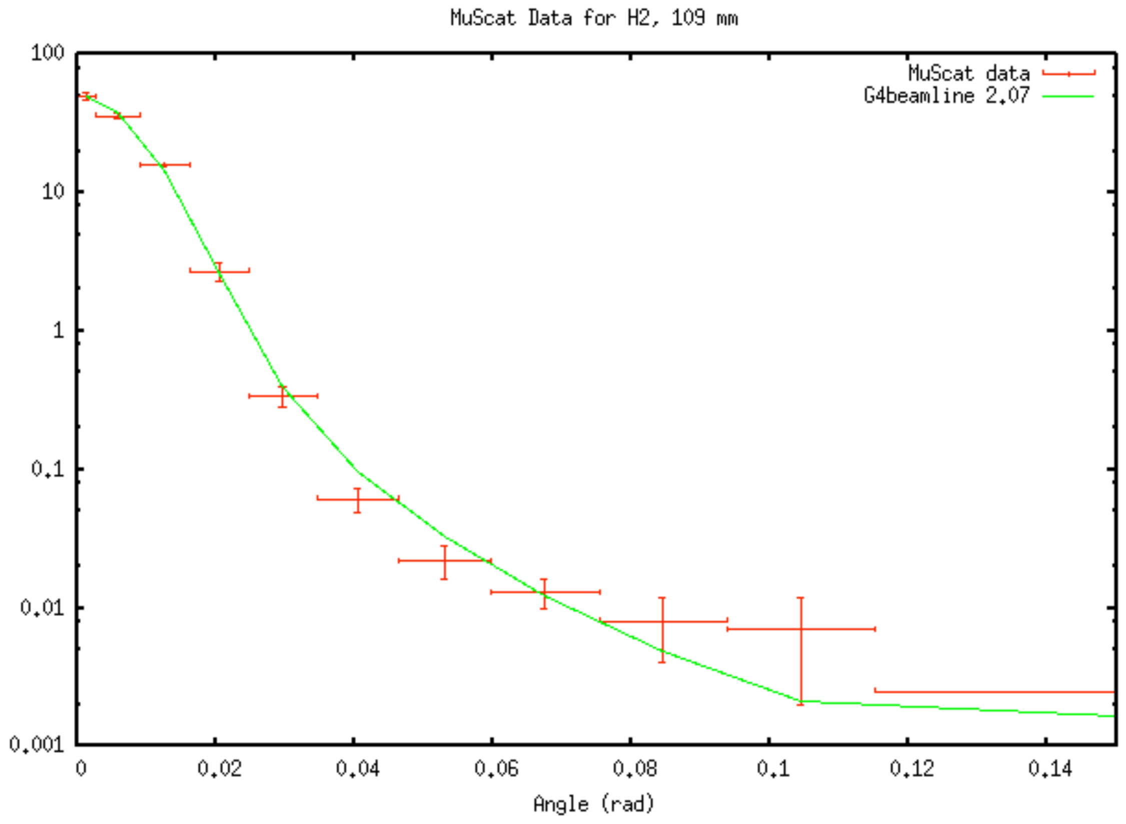
Test 2
A second test is to check that particles move in straight lines through vacuum with no fields. A proton beam with zero emittance and meanXp=0.000001 (i.e. dx/dz = 1 microradian) has every particle at y=0.0 and x=10000.0 at z=1E10, to the accuracy of a float (tracking is performed in double precision, but the NTuple uses float-s). This test has the default maxStep=100 (mm), and took ~800 seconds to track each particle for 10,000 km (100 million steps).

3.2 Electromagnetic Interactions

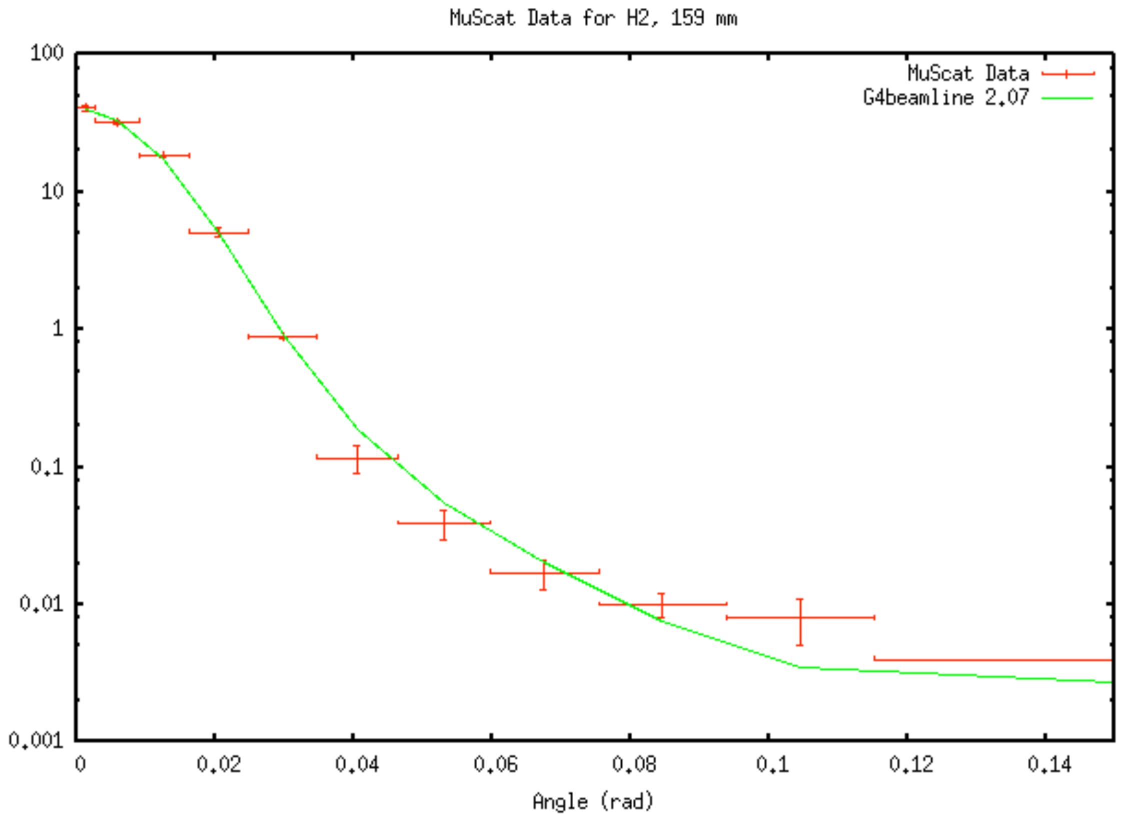
The primary link for the Geant4 Collaboration's testing and validation of electromagnetic physics processes is: https://twiki.cern.ch/twiki/bin/view/Geant4/EMValidation. A large body of work is reported there.

3.2.1 Multiple Scattering

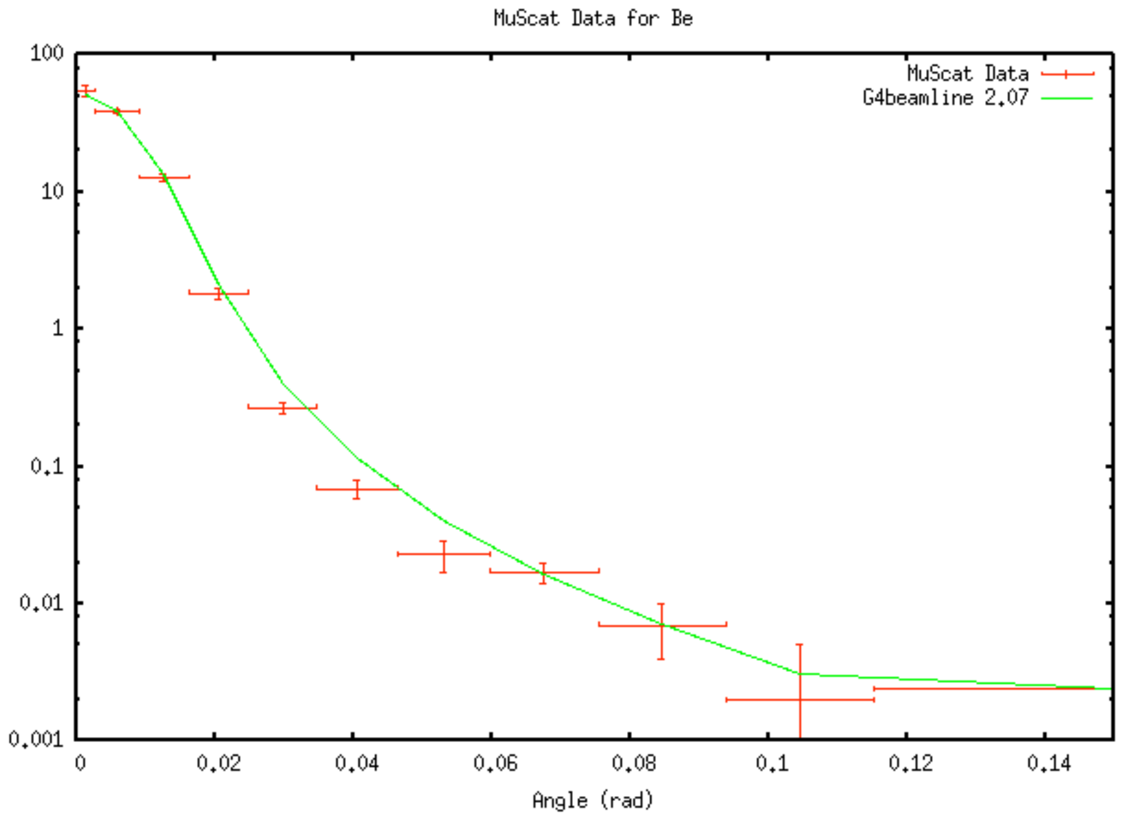
Test1 - MuScat
The MuScat experiment [6] ran at TRIUMF to measure μ^+ multiple scattering in various materials. It was inspired by the need for accurate data to validate simulation programs like G4beamline, specifically for muon cooling studies. It used a 172 MeV/c μ^+ beam on several different targets. For liquid hydrogen, the experimenters de-convolved the data using target-empty runs, to remove the effects of the windows. The G4beamline runs use 10 million events each, and the errorbars from G4beamline are negligible compared to those from MuScat.

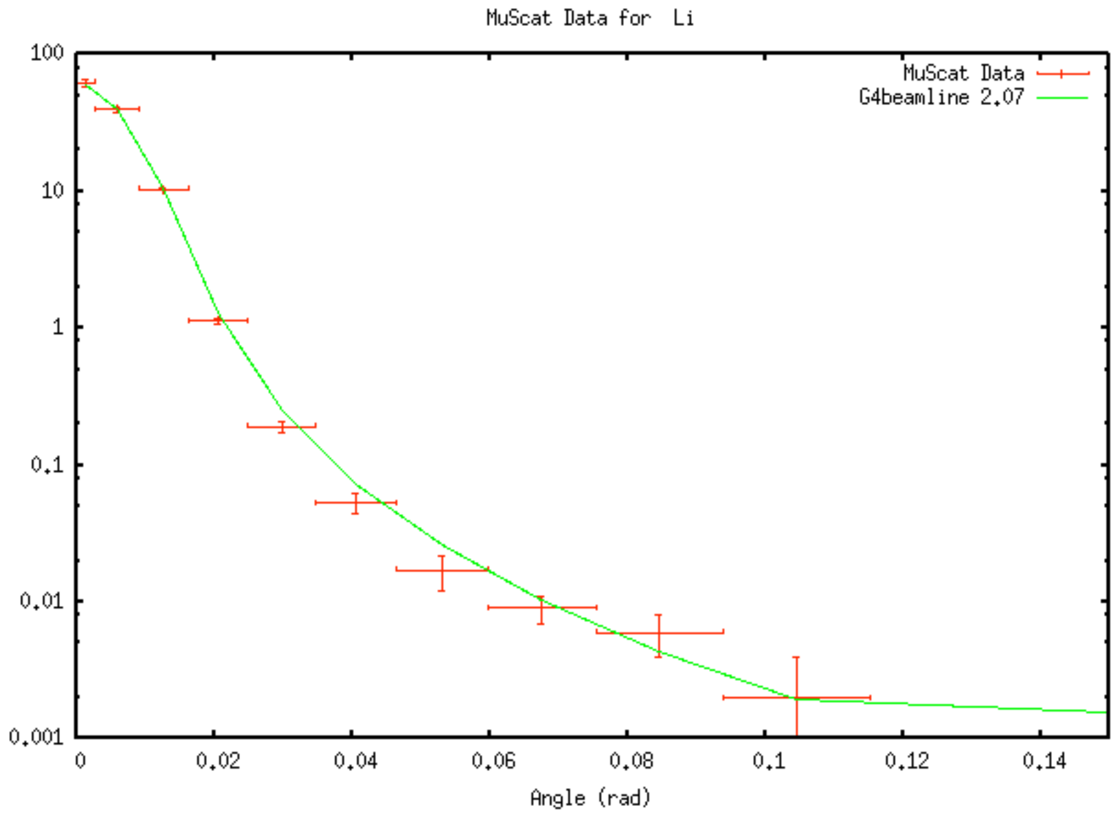


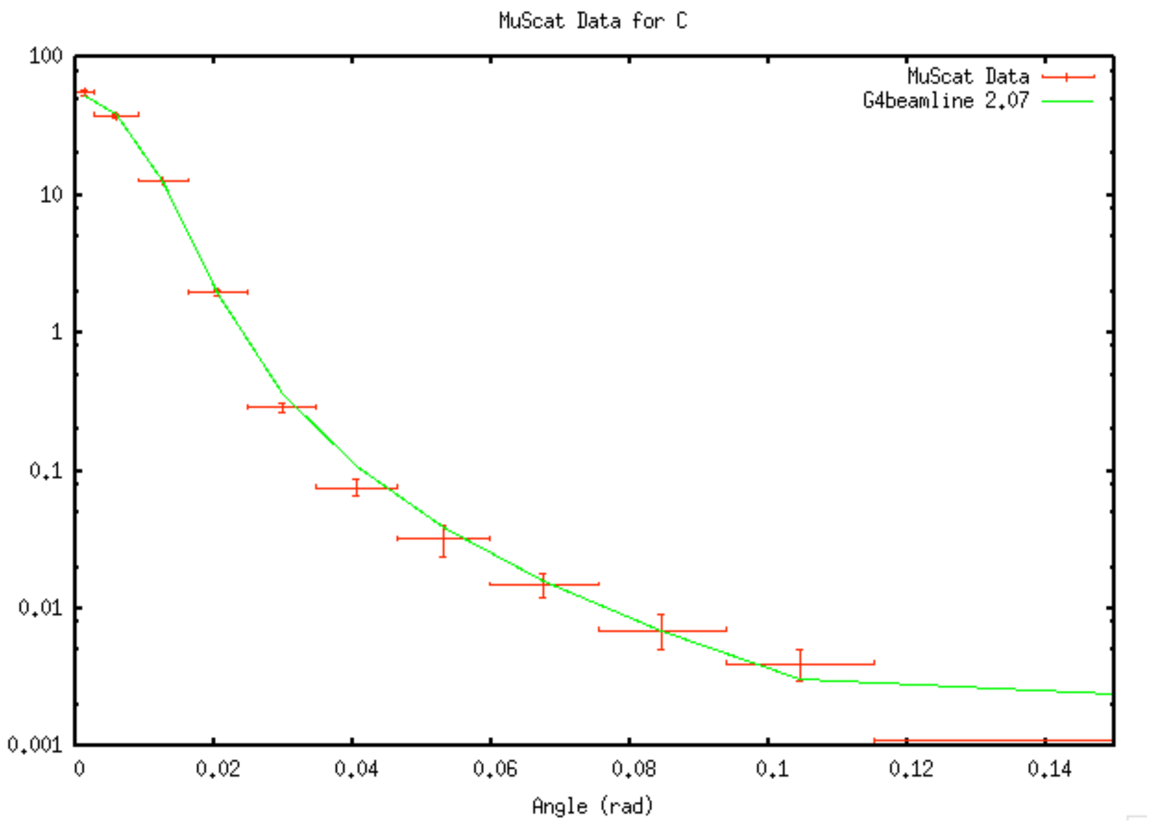
MuScat Data for H2, 109 mm
MuScat data
G4beamline 2.07
100
10
1
0.1
0.01
0.001
0
0.02
0.04
0.06
0.08
0.1
0.12
0.14
Angle (rad)

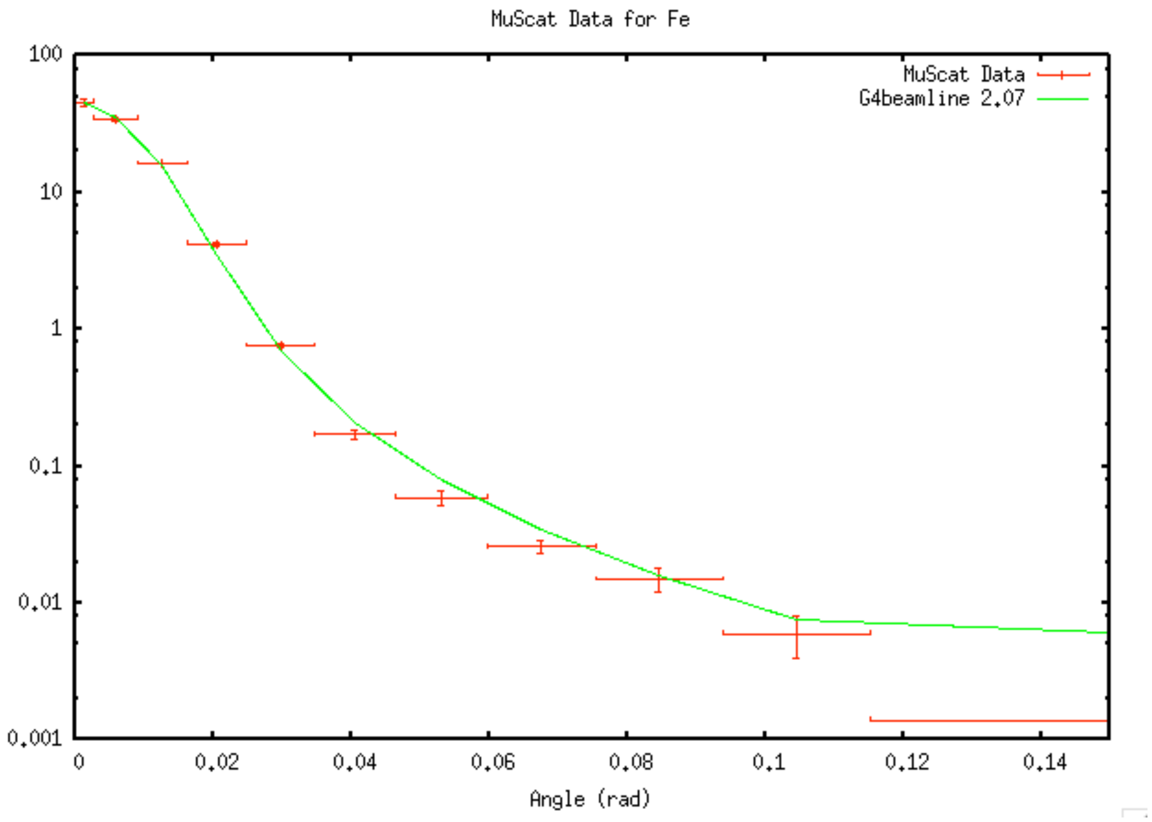


MuScat Data for H2, 159 mm
MuScat Data
G4beamline 2.07
100
10
1
0.1
0.01
0.001
0
0.02
0.04
0.06
0.08
0.1
0.12
0.14
Angle (rad)


MuScat Data for Be
MuScat Data
G4beamline 2.07
100
10
1
0.1
0.01
0.001
0
0.02
0.04
0.06
0.08
0.1
0.12
0.14
Angle (rad)


MuScat Data for Li
MuScat Data
G4beamline 2.07
100
10
1
0.1
0.01
0.001
0
0.02
0.04
0.06
0.08
0.1
0.12
0.14
Angle (rad)


MuScat Data for C
MuScat Data
G4beamline 2.07
100
10
1
0.1
0.01
0.001
0
0.02
0.04
0.06
0.08
0.1
0.12
0.14
Angle (rad)


MuScat Data for Fe
MuScat Data
G4beamline 2.07
100
10
1
0.1
0.01
0.001
0
0.02
0.04
0.06
0.08
0.1
0.12
0.14
Angle (rad)

While these plots look reasonable, and certainly reproduce the broad features of the data, when examined quantitatively the correspondence is not so good. The G4beamline data can be considered as a zero-parameter fit to the MuScat data with 11 degrees of freedom, and the corresponding Chisq values are given below. Note the bins for the two datasets are identical, so the errorbars in x have no effect (the lines for G4beamline data above are drawn point-to-point at the same bin centers as the MuScat data).

Target	**Chisq**	**Chisq/DF**	**Probability**
H2 (109 mm)	28.8	2.62	0.3%
H2 (159 mm)	47.1	4.28	<0.01%
Be	63.7	5.79	<0.01%
Li	27.6	2.51	0.4%
C	27.0	2.45	0.4%
Fe	87.5	7.96	<0.01%

For all six targets, the largest contributions to Chisq come from bins in the range 0.02 < Angle < 0.05. These are neither in the center nor in the tail. In all cases, at least 1/3 of the Chisq comes from a single bin; for the three worst targets, more than half of the Chisq comes from a single bin.

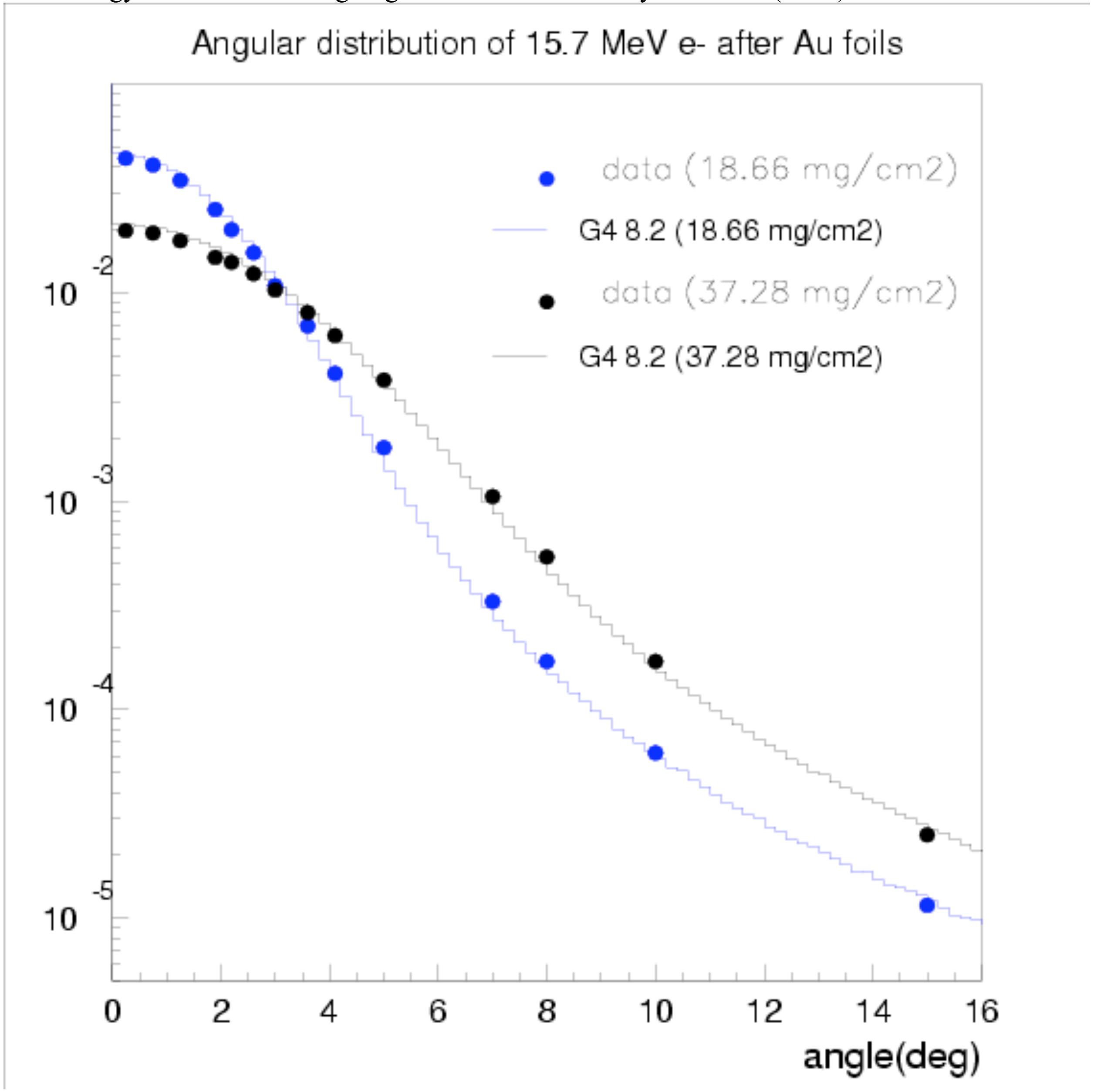
The input file for these tests is muscat.g4bl:

```
*	Muscat. in  TJR  1-FEB-2006  mu+ scattering
#	UPDATED 29-NOV-2010 TJR
#
#	lengths are mm; momentum is MeV/c, density is gm/cm^3
#
physics QGSP_BIC
beam gaussian particle=mu+ meanMomentum=172 nEvents=10000000
trackcuts keep=mu+
#
#material Li A=6.941 Z=3 density=0.53
#tubs Target outerRadius=100 material=Li length=12.78 color=1,0,0
#
## material is Be2 because Be is already defined with slightly different density
#material Be2 A=9.012182 Z=4 density=1.85
#tubs Target outerRadius=100 material=Be2 length=3.73 color=1,0,0
#
#material H2 A=1.00794 Z=1 density=0.0755
#tubs Target outerRadius=100 material=H2 length=109 color=1,0,0
#tubs Target outerRadius=100 material=H2 length=159 color=1,0,0
#
#material C A=12.011 Z=6 density=1.69
#tubs Target outerRadius=100 material=C length=2.5 color=1,0,0
#
material Fe A=55.845 Z=26 density=7.86
tubs Target outerRadius=100 material=Fe length=0.24 color=1,0,0
```

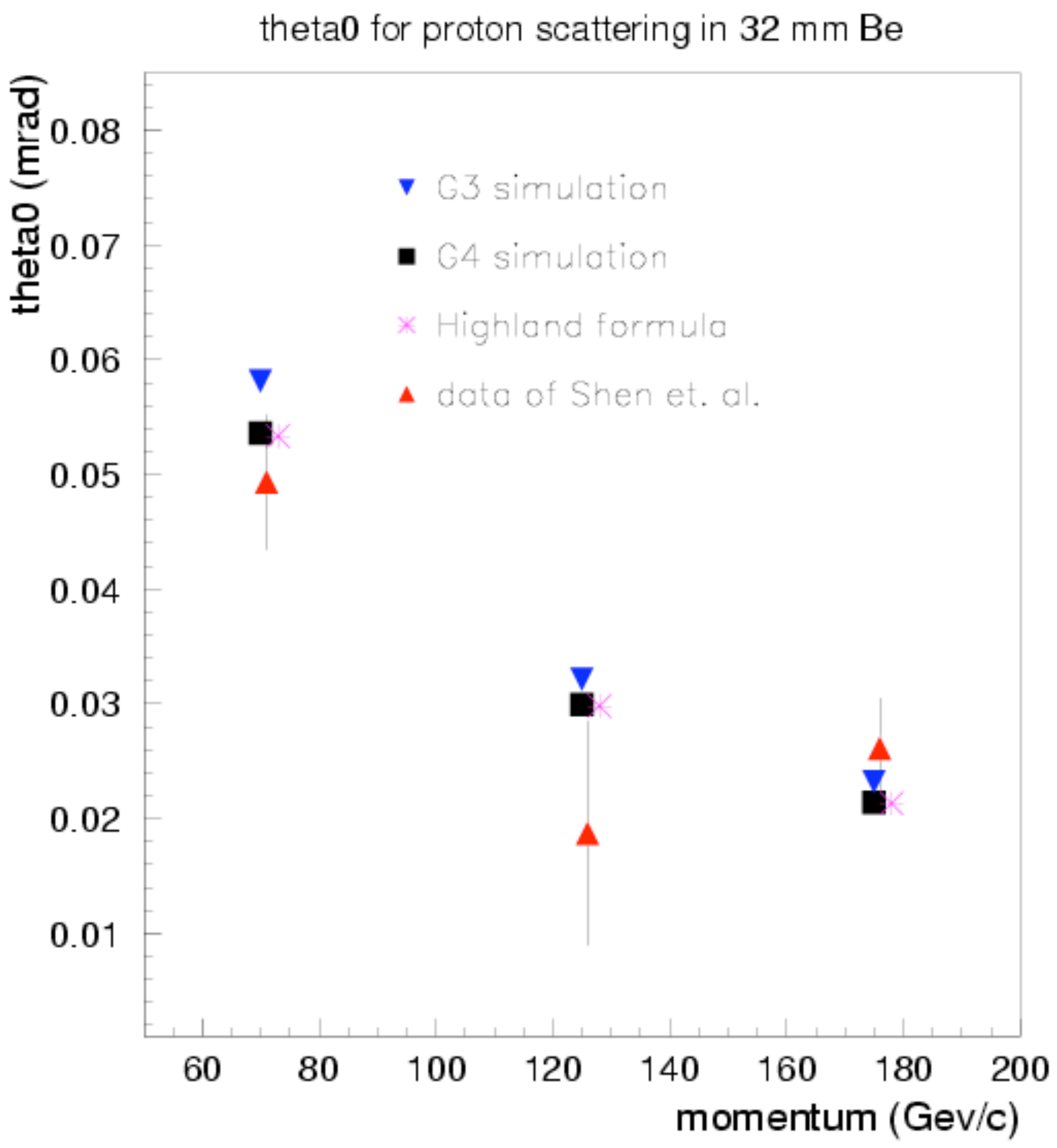
```
#
virtualdetector Det radius=10000 color=0,1,0 format=ascii
place Target z=100
place Det z=200
```

Test2 – Geant4 Collaboration tests
The Geant4 electromagnetic physics working group page on testing and validation is located at:
http://geant4.web.cern.ch/geant4/collaboration/working_groups/electromagnetic/tests.shtml
It contains considerably more tests than just multiple scattering. Two plots from the section on multiple scattering are reproduced below.

Low energy electron scattering in gold. Data are from Phys. Rev. 84 (1951) 634.



High-energy proton scattering in Be. Data are from Phys. Rev. D 20 (1979) 1584.

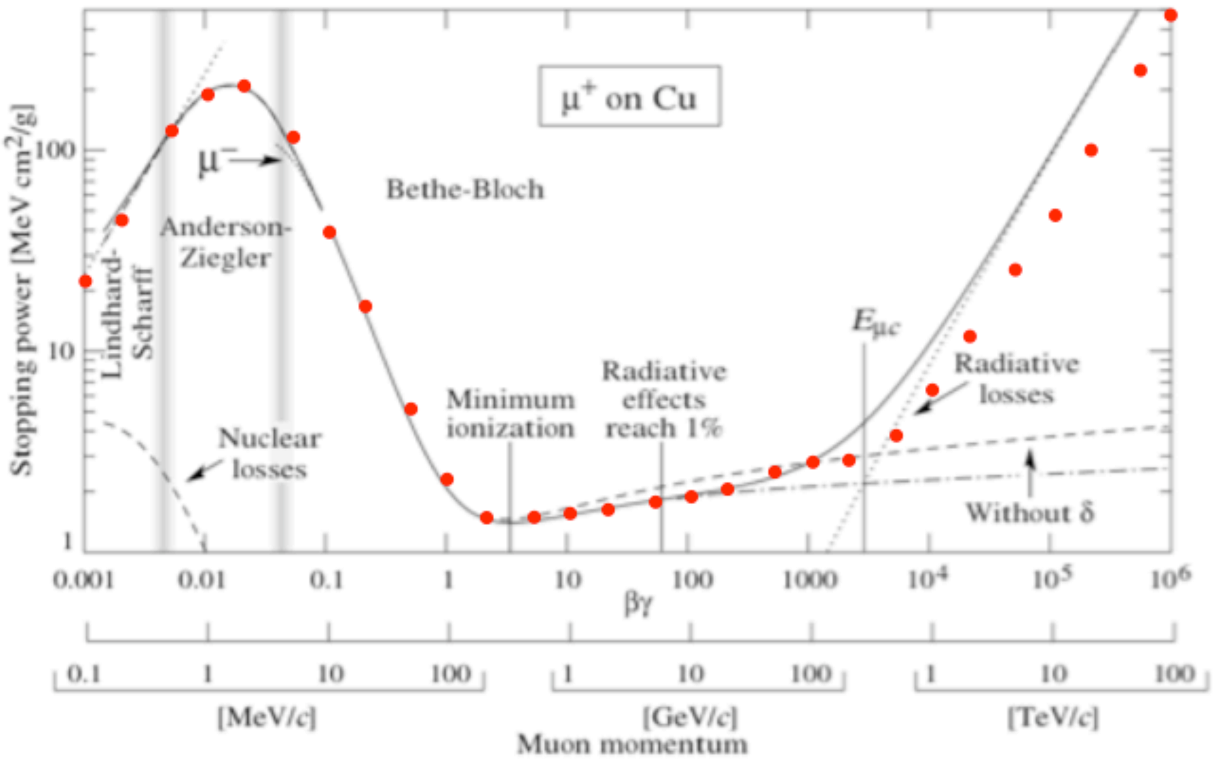


3.2.2 Ionization Energy Loss

Test1 Comparison to the PDG model

The ionization energy loss in Geant4 for μ^+ tracks well with the PDG model below $\beta\gamma \sim 1000$. Above that, the discrepancy is claimed to be due to nuclear absorption of muons, which still needs to be investigated.

The following plot is from the PDG “Review Tables and Plots” section on “Passage of particles through matter” [7], fig. 27.1. The red dots were computed using G4beamline 1.16; errorbars are smaller than the dots. For each point, the thickness of the Cu absorber was varied so the incident μ^+ lose about 5% of their momentum.



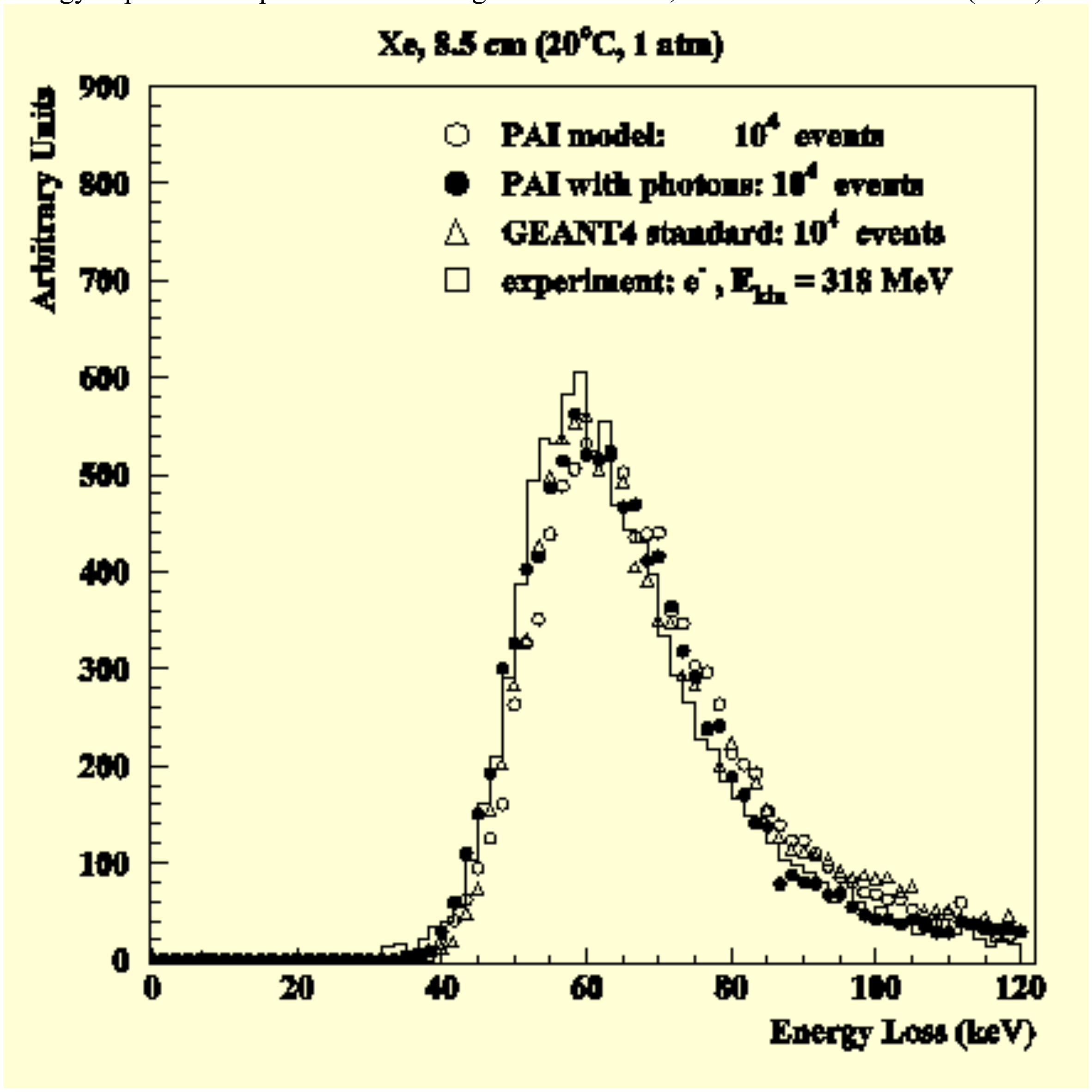
The input file for this test is eloss.g4bl:

```
#       eloss.g4bl
#       vary Thick so mu+ lose about 5% of their initial momentum.
param -unset P=1.0 Thick=1.0 Det=1.0.txt
param material=Cu M=105.658
physics QGSP_BERT disable=Decay
trackcuts kill=e+,e-,gamma
beam gaussian particle=mu+ meanMomentum=$P nEvents=10000
tubs Absorber outerRadius=200 material=$material length=$Thick
place Absorber z=1000
virtualdetector Det radius=200 length=1 format=ascii file=$Det
place Det
```

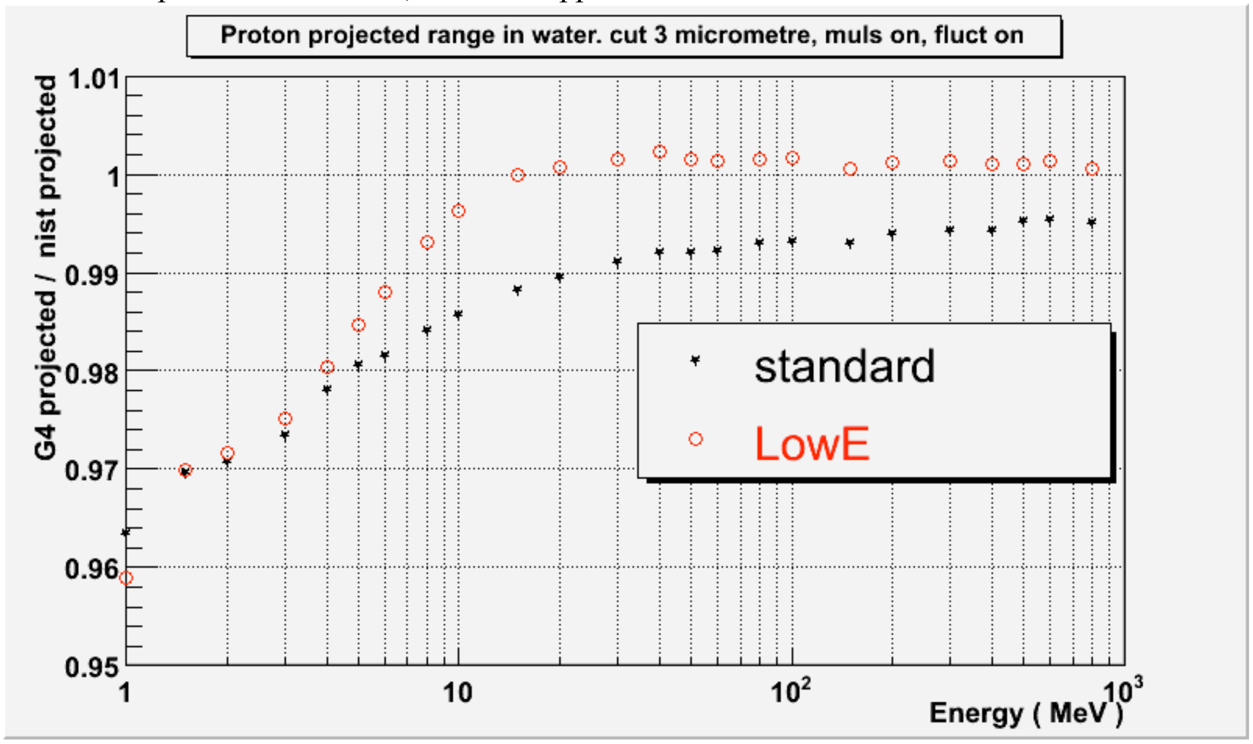
Test2 Geant4 Collaboration tests

The Geant4 electromagnetic physics working group page on testing and validation is located at:
http://geant4.web.cern.ch/geant4/collaboration/working_groups/electromagnetic/tests.shtml
It contains considerably more tests than just ionization energy loss. A few figures are copied below.

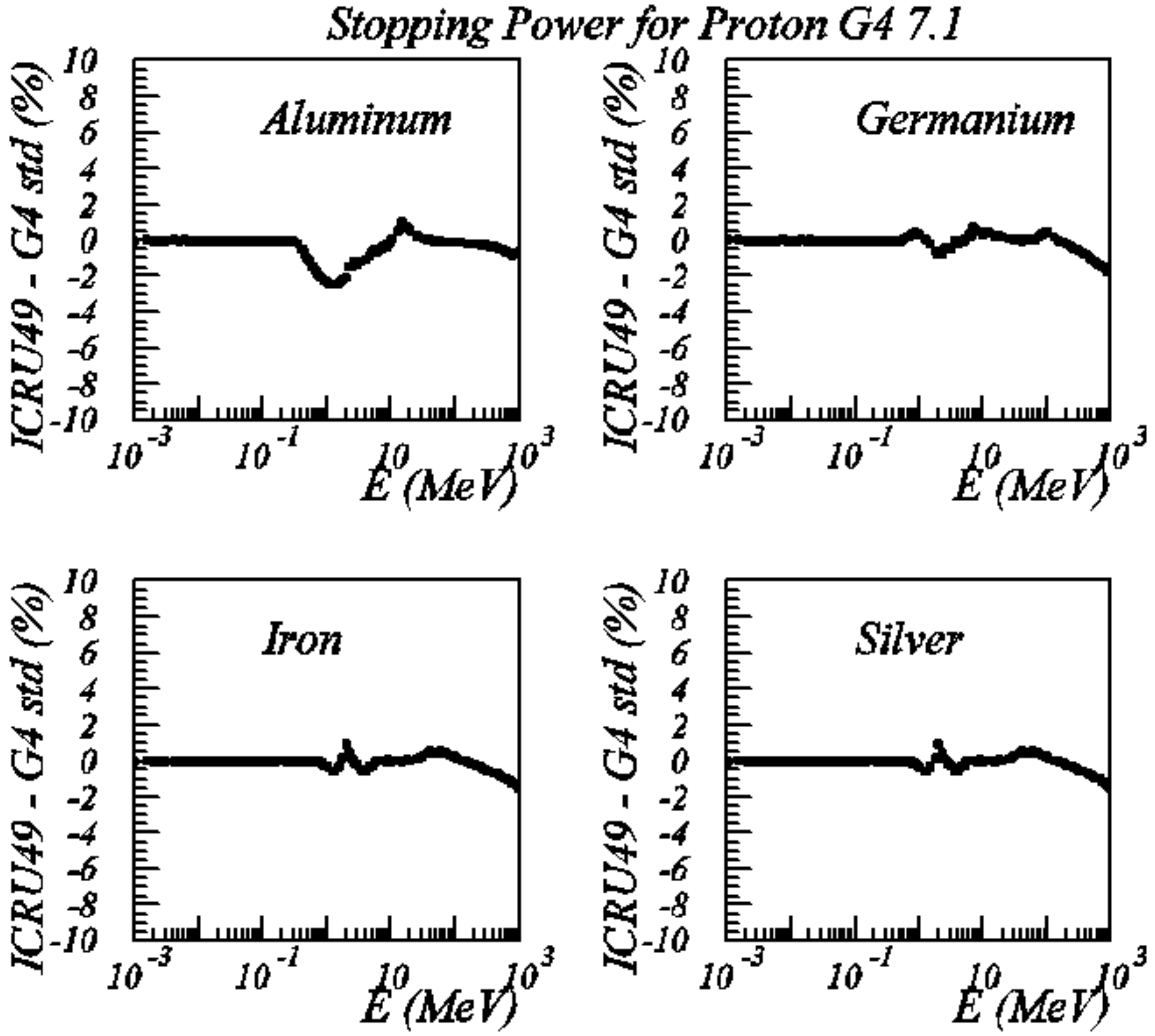
Energy deposition of protons in Xenon gaseous detector; data are from NIM 217 (1983) 277.



Proton projected range, ratio of the Geant4 calculation to the NIST data, as a function of energy. Accuracy of the data above 1 MeV is claimed to be about 2%. This plot shows the ratio of the Geant4 computation to the data; note the suppressed zero.



Proton stopping power versus ICRU'49 evaluated data for Geant4 7.1p01. The accuracy of the data above 1 MeV is claimed to be about 2%. These plots show the difference in percent between the data and the Geant4 computation.



3.2.3 Energy Straggling

The following plot is from Tschalär and Maccabee, Phys. Rev. B 1, no. 7, p2863 (1970). It shows the straggling of 19.68 MeV protons after various thicknesses of Al absorber. The red dots are G4beamline. The G4beamline beam has zero energy width, which is lower than that of their beam (~0.03 MeV, comparable to their detector resolution), but this is not enough to account for the differences. Clearly, the Geant4 energy-loss process loses slightly too little energy, and has too little straggling. This simulation has four absorbers in a row so all four plots could be made at once, but the difference between that and a single absorber of 0.497 g/cm^2 is barely visible.

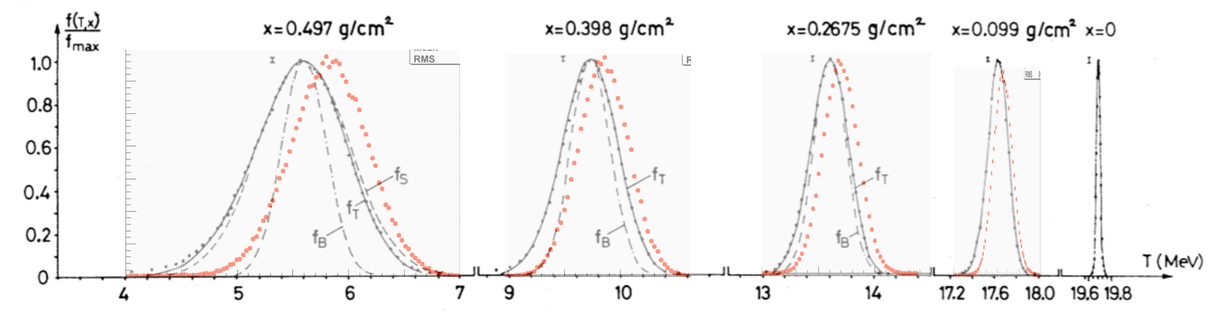
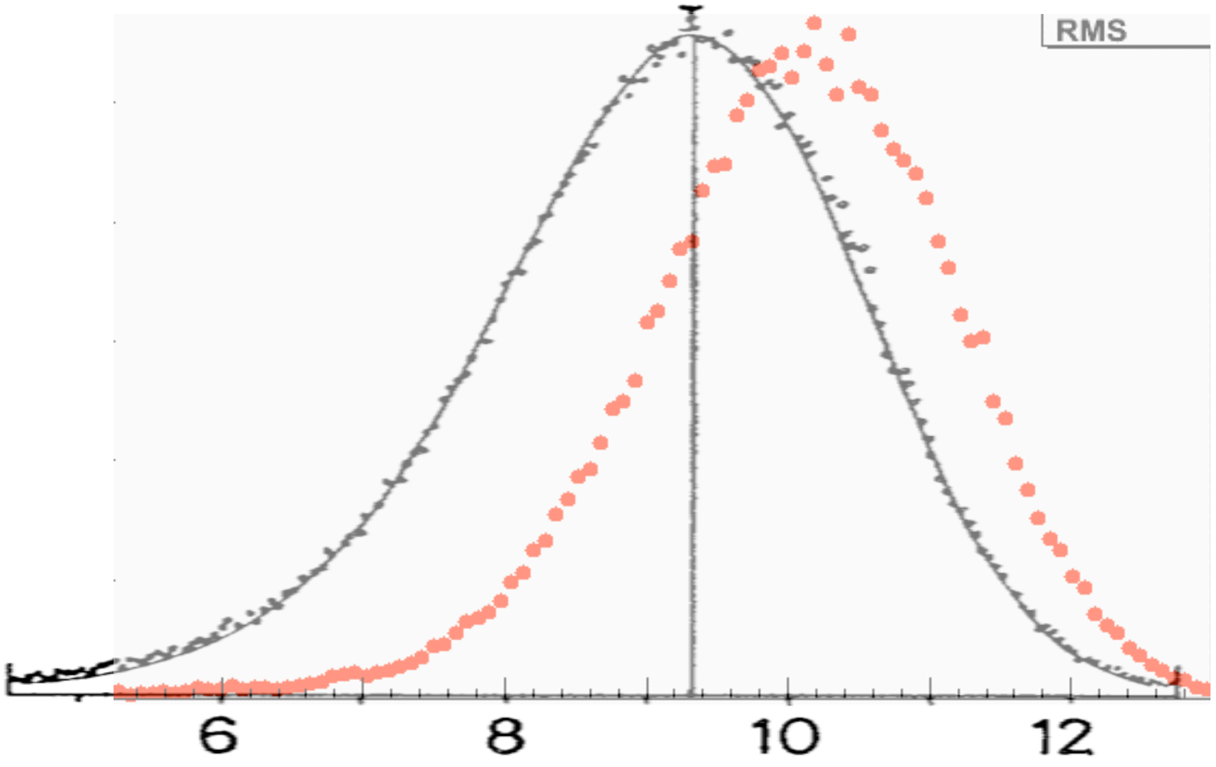


FIG. 3. Energy distributions of 19.68-MeV protons after traversing an aluminum absorber. Curves f_B are theoretical predictions of Bohr, f_S of Symon, and f_T of Payne and Tschalär.

The same difference applies to 49.10 MeV protons after a thicker Al absorber (2.675 g/cm^2); data are from the same reference as above. The peak energy loss differs by about 1 MeV, out of 40 MeV in energy loss, a difference of 2.5%.



The input file for this test is straggling.g4bl:

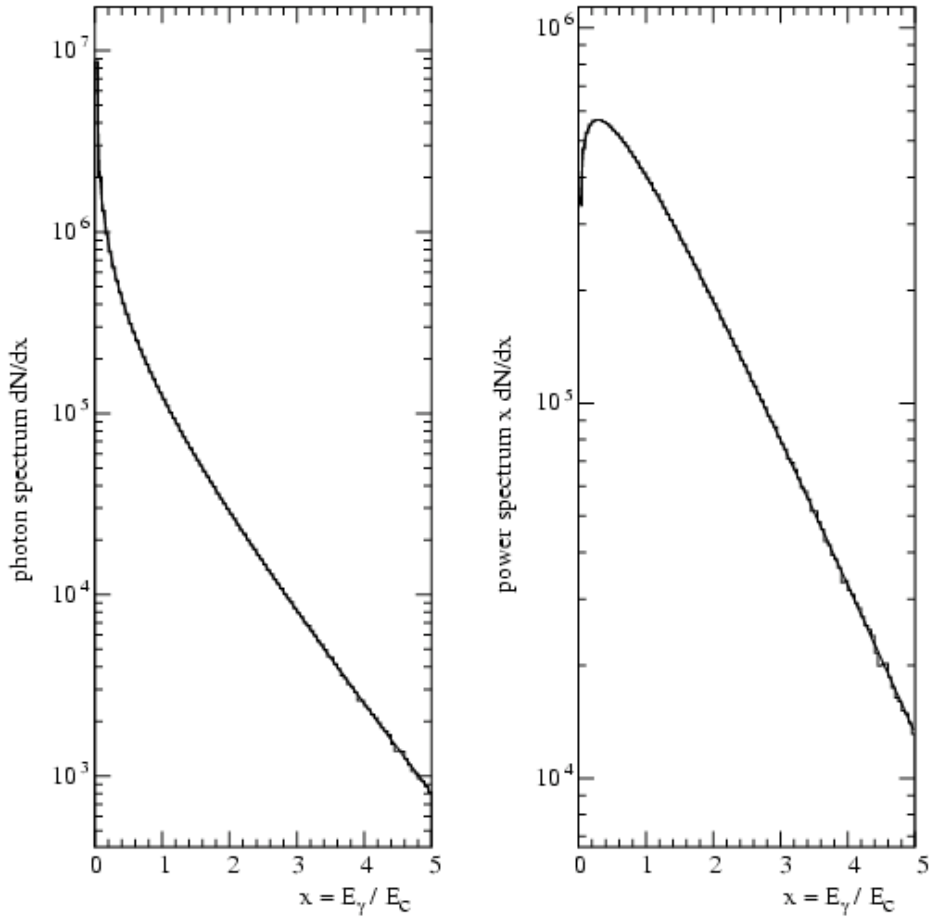
```
#        straggling.g4bl
physics QGSP_BERT
param DEN=2.699 M=938.272 KE=19.68
param P=sqrt(($KE+$M)*($KE+$M)-$M*$M)
beam gaussian particle=proton meanMomentum=$P nEvents=100000
virtualdetector Det radius=1000 length=0.01 color=0,0,1
place Det z=1 rename=0
```

```
cylinder Abs1 material=Al outerRadius=1000 length=0.099/$DEN*10
place Abs1 z=$Zcl+10
place Det rename=0.099 z=$Zcl+10
cylinder Abs2 material=Al outerRadius=1000 length=(0.2675-0.099)/$DEN*10
place Abs2 z=$Zcl+10
place Det rename=0.2675 z=$Zcl+10
cylinder Abs3 material=Al outerRadius=1000 length=(0.398-0.2675)/$DEN*10
place Abs3 z=$Zcl+10
place Det rename=0.398 z=$Zcl+10
cylinder Abs4 material=Al outerRadius=1000 length=(0.497-0.398)/$DEN*10
place Abs4 z=$Zcl+10
place Det rename=0.497 z=$Zcl+10
```

3.2.4 Synchrotron Radiation and Other EM Processes

The Geant4 Collaboration test of synchrotron radiation is in:
http://geant4.web.cern.ch/geant4/collaboration/working_groups/electromagnetic/tests.shtml#SR.
The plot is copied below.

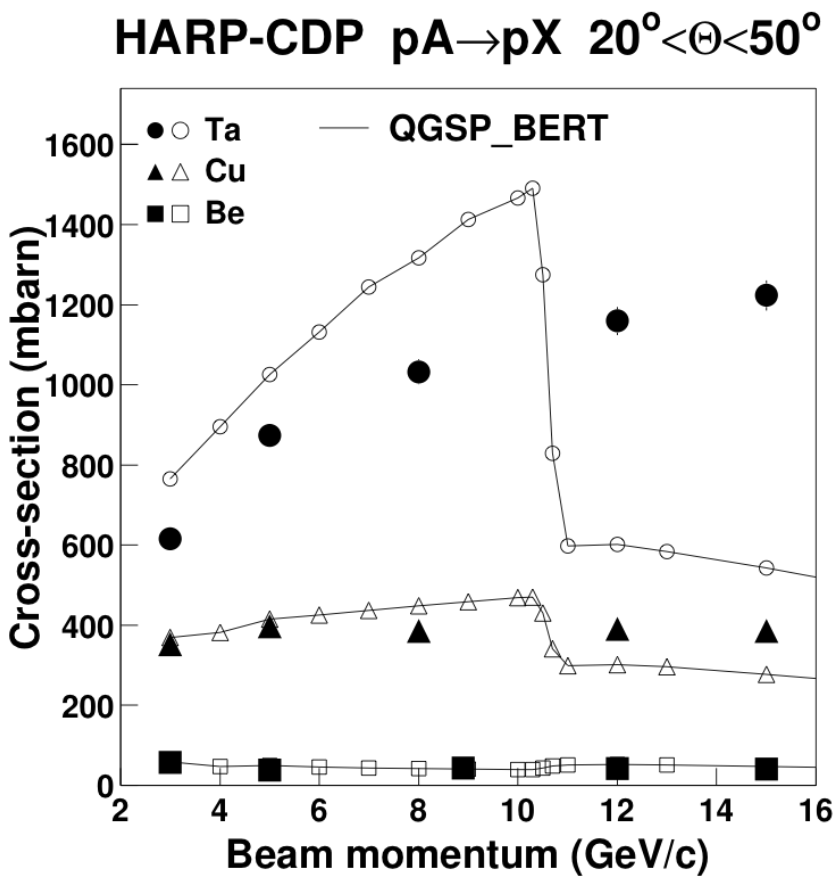
Synchrotron radiation: comparison of theoretical (smooth curve) and generated (histogram) spectra.



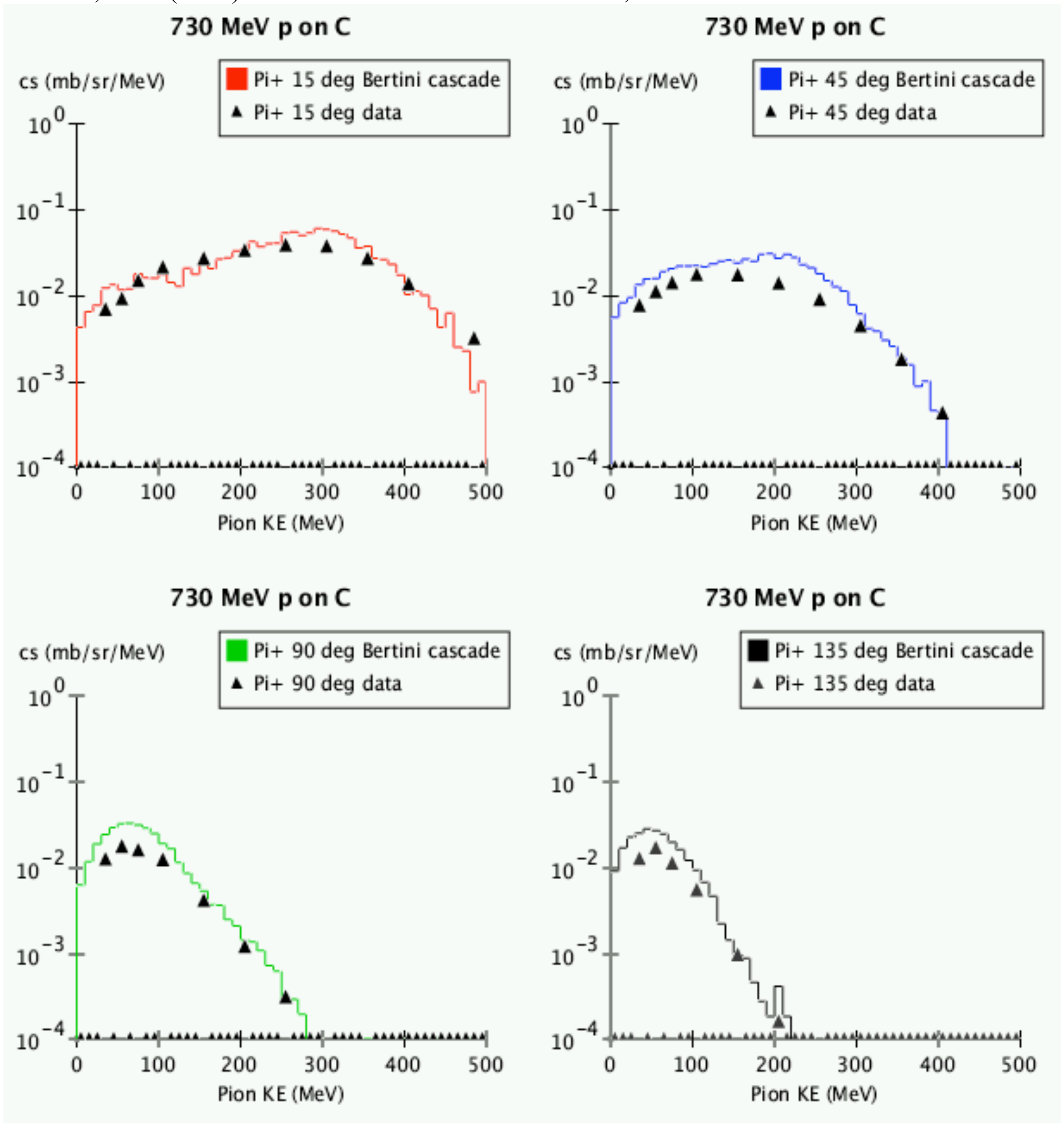
3.3 Hadronic Interactions

The primary link for the Geant4 Collaborations webpage on "Physics Validation and Verification" for hadronic physics processes is: http://geant4.web.cern.ch/geant4/results/validation_plots.htm. A very tiny sample of representative plots is copied here.

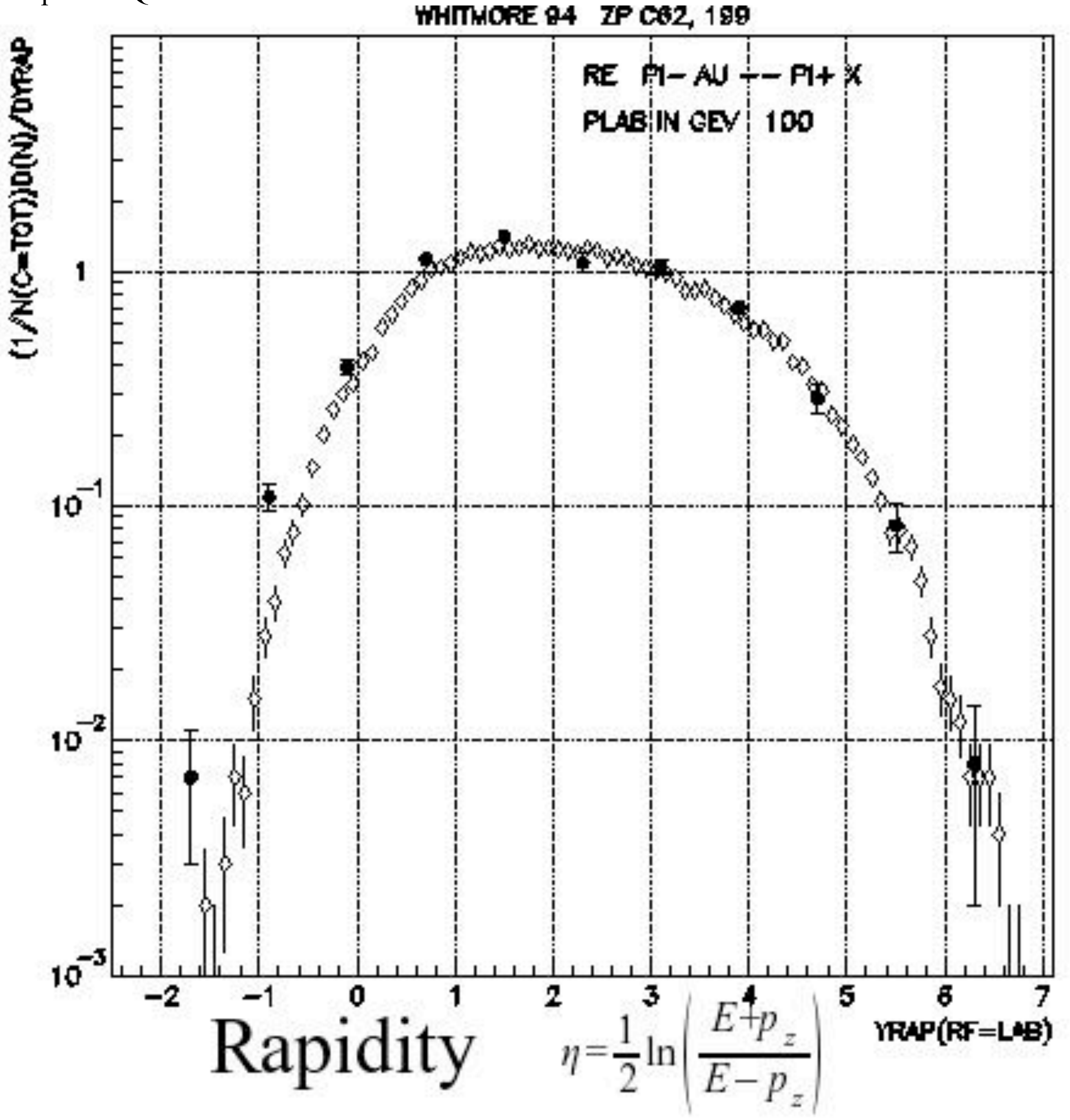
Comparison to HARP inclusive proton production data, p A → p X, 20° < angle < 50°. The transition region of QGSP_BERT just above 10 GeV is quite evident. Plot is from arXiv:1006.3429.



A proton beam of 730 MeV is incident upon a C target. The double-differential pi+ cross sections are measured vs. angle and kinetic energy. Data are from: D.R.F. Cochran et al., Phys. Rev. D6, 3085 (1972). Error bars are between 8 and 18%, statistical.



A 100 GeV/c pi- beam is incident upon an Au target. Rapidity distributions of the final state pi+ are plotted. Data are from: J.J. Whitmore et al., Z. Phys. C62, 199 (1994). Filled points: data, open points: QGS model.



3.4 Weak Interactions

The Geant4 Collaboration does not report on validation of weak interaction processes.

3.5 Optical Photons

Optical photons are not yet implemented in G4beamline.

4 Physics Processes Implemented in G4beamline

4.1 Pion Decay (updated from Geant4)

By default, Geant4 does not include the rare decay modes of charged pions:

$\pi^+ \rightarrow e^+ \nu_\mu$ (1.230E-4)

$\pi^- \rightarrow e^- \overline{\nu}_\mu$ (1.230E-4)

G4beamline adds these modes. The other rare-decay modes have not been implemented, due to the complexity of modeling 3- and 4-body decays; they are a factor of 1000 (or more) below the above. Moreover, unlike these modes, they do not generate additional backgrounds for the Mu2E experiment [8], which is the primary customer for this.

The *setdecay* command can be used to modify particles' decay modes, including changing their lifetime, adding and removing specific modes, and changing their branching ratios (e.g. for testing background rejection, pions could be forced to always decay into these normally rare modes).

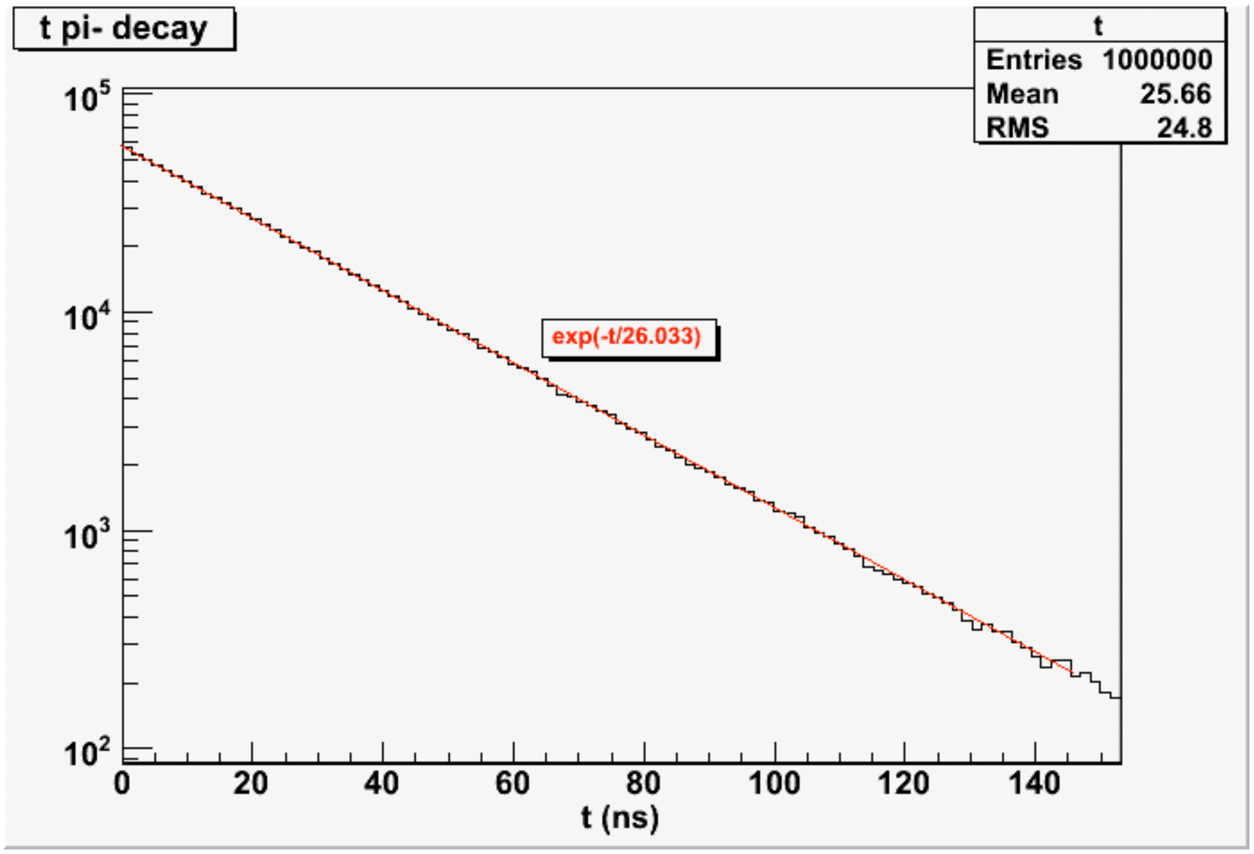
The decay products from 1,000,000 π^+ decays, with μ^+ decays inhibited, are:

```
-13  999875  mu+
-11     125  e+
 12     125  nu_e
 14  999875  nu_mu
```

The decay products from 1,000,000 π^- decays, with μ^- decays inhibited, are:

```
-14  999886 anti_nu_mu
-12     114 anti_nu_e
 11     114 e-
 13  999886 mu-
```

The lifetime of π^- is correct (26.033 ns):

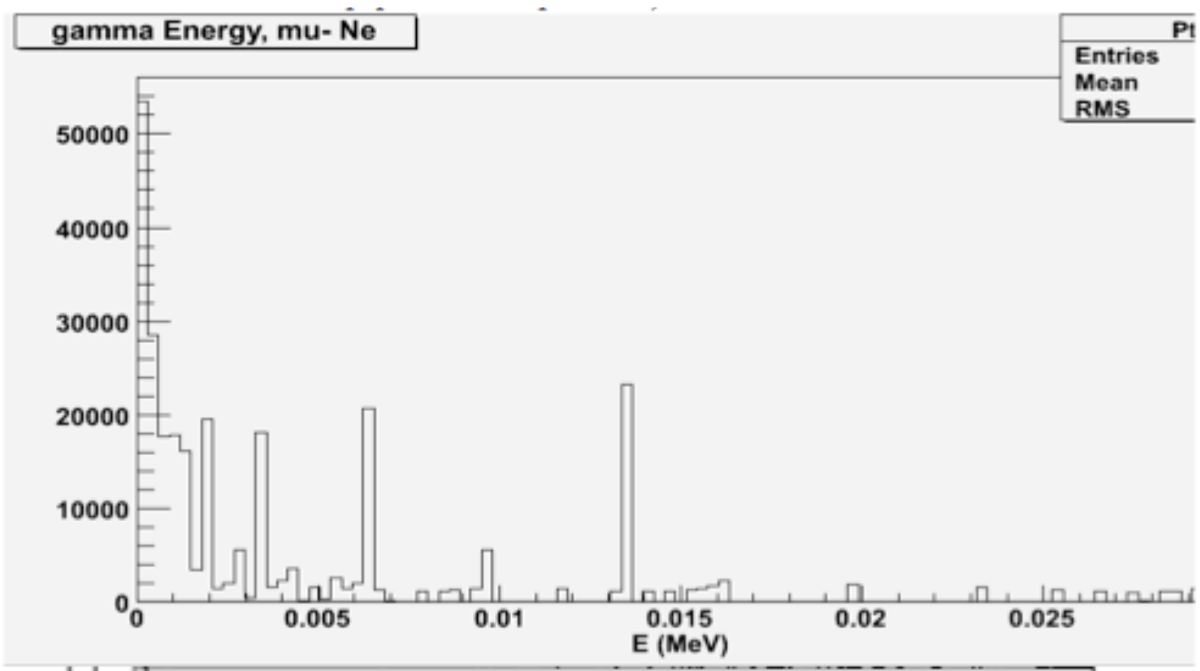


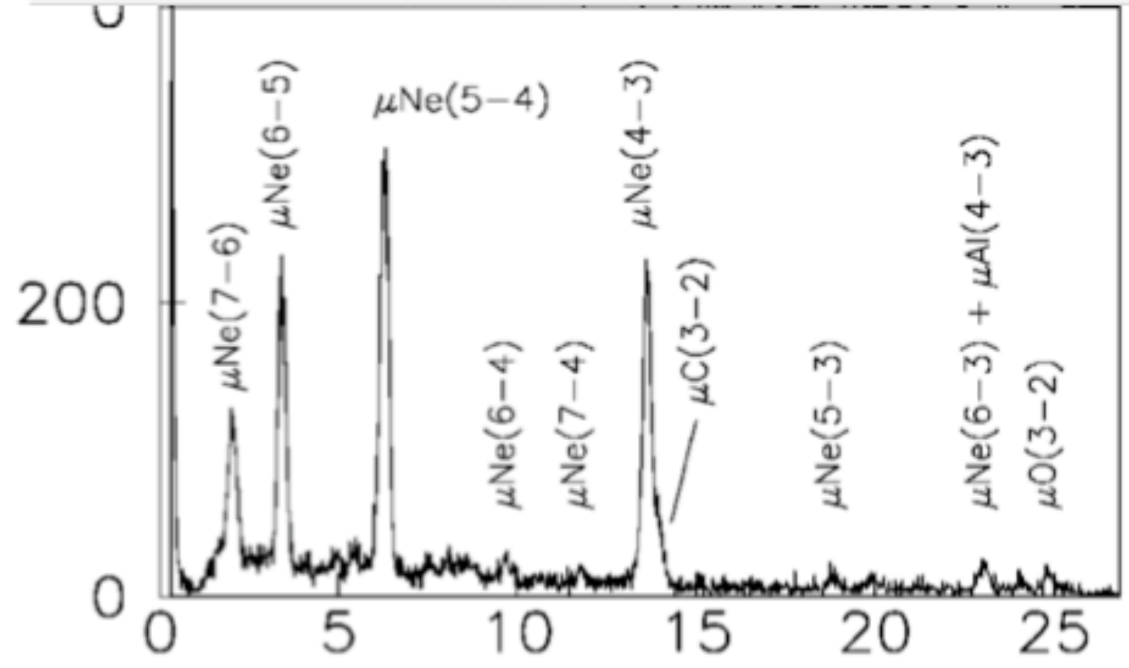
The input file for this test is decay.g4bl:

```
*       decay.g4bl - test pion decay
randomseed time
physics QGSP_BERT
setdecay mu+ lifetime=1E99
beam gaussian particle=pi+ meanMomentum=10 nEvents=1000000
newparticlentuple New
box End height=1 width=1 length=1
place End z=100000
```

4.2 Mu- Capture (updated from Geant4)

As mu- capture at rest is very important to the Mu2E experiment [8], considerable testing has been performed. See the report "Analysis of Geant4 Physics Processes for μ^- Capture at Rest" [9]. The X-Ray spectrum from neon is reproduced well, as shown in the following figure:

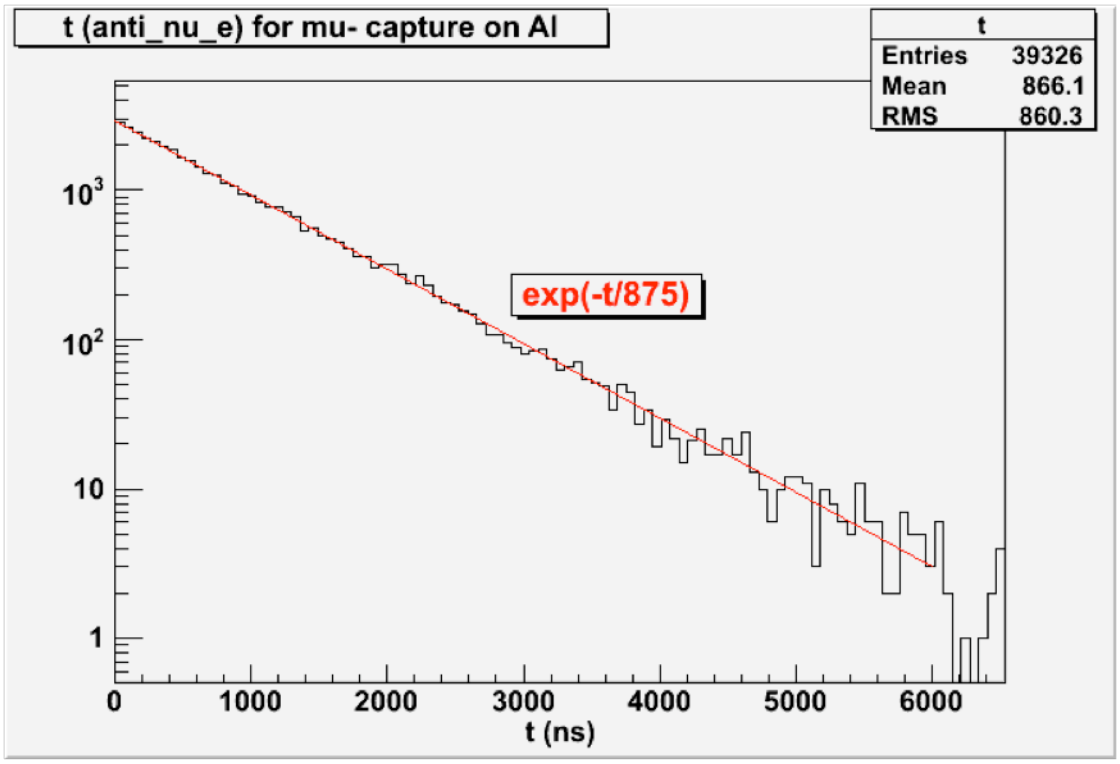




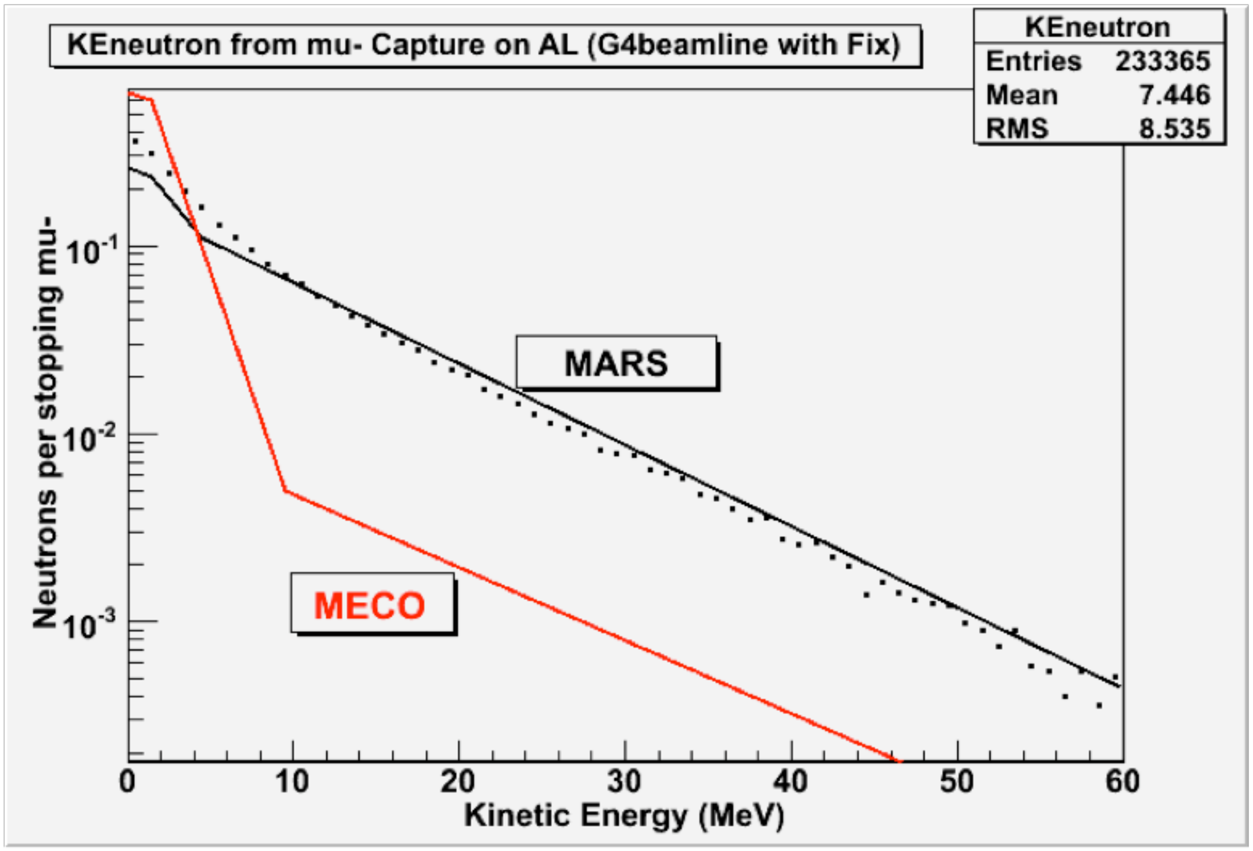
Experimental X-ray Spectrum for μ^- Neon, from Kirch *et al*, Phys. Rev. **A59**, p3375.

Agreement with the experiment is good, except for an excess below 1 keV where their detector loses efficiency.

The lifetime for decay in orbit is in good agreement with measurements:



The generation of neutrons from nuclear captures of mu- was found to be incorrect, missing a high-energy tail. A new command *muminuscapturefix* was implemented to add the missing neutrons with the distribution from MARS [10]. This is a stopgap fix to permit the investigation of neutron backgrounds, and is simply adding isotropic neutrons with an exponential energy distribution fit to the MARS high-energy tail. In the following plot, the lines are for MARS and the MECO simulation; the dots are for G4beamline (available experiments have very poor resolution in this region).



4.3 Bug 1021 (update to Geant4 tracking in E field)

Bug 1021 was reported to the Geant4 developers on 20-AUG-2008; at present, it has been assigned but not resolved (it is a rather difficult problem, as it is related to the fundamental design of Geant4 tracking). This bug only affects very low energy charged particles that travel anti-parallel to an E field, stop, and turn around. If the track is just a few degrees from being anti-parallel to E, the tracking is OK, but for tracks that do stop, the results can be grotesquely wrong (huge kinetic energies, outrageous time delays). The basic problem is that Geant4 tracks in space, which becomes singular at points where the particle velocity is zero. G4beamline has a workaround for this bug, the *bug1021* command, which applies a variable step size as the turn-around is approached, and "jumps" through the turnaround when it is less than 2 microns away. Note that for negative particles in a material with an E field, the appropriate atomic capture-at-rest process is not invoked.

4.4 Collective Computations

Geant4 is designed and intended for single-particle simulations. But beams inherently have many particles, and in some cases, collective effects are important. G4beamline has an infrastructure that supports collective computations; it is used in the computations of space charge.

Implementing collective computations requires a completely new RunManager, EventManager, TrackManager, and SteppingManager in the Geant4 tracking code (the original versions are used when collective mode is not invoked, so comparisons can be made easily). The idea is to track all particles essentially in parallel, for steps in time (not space). So every particle is given a new physics process that will suspend it when it reaches the designated step time (it uses the particle's current velocity to limit the step in space). The algorithm used is:

1. Create a vector of the initial (beam) tracks.
2. Find the largest global time in all tracks, and set the step time to it.
3. Loop over the track vector, stepping each active track until it is killed or suspended (at the step time).
4. Call the collectiveComputation() function with the current track vector.
5. Increment the step time by the current value of deltaT.
6. Change all tracks with status=suspended to status=active.
7. Loop back to item 3 until no tracks are active.

The user sets the initial value of deltaT (the time increment between steps); the collectiveComputation() function can modify it, if necessary, depending on the detailed needs of each computation.

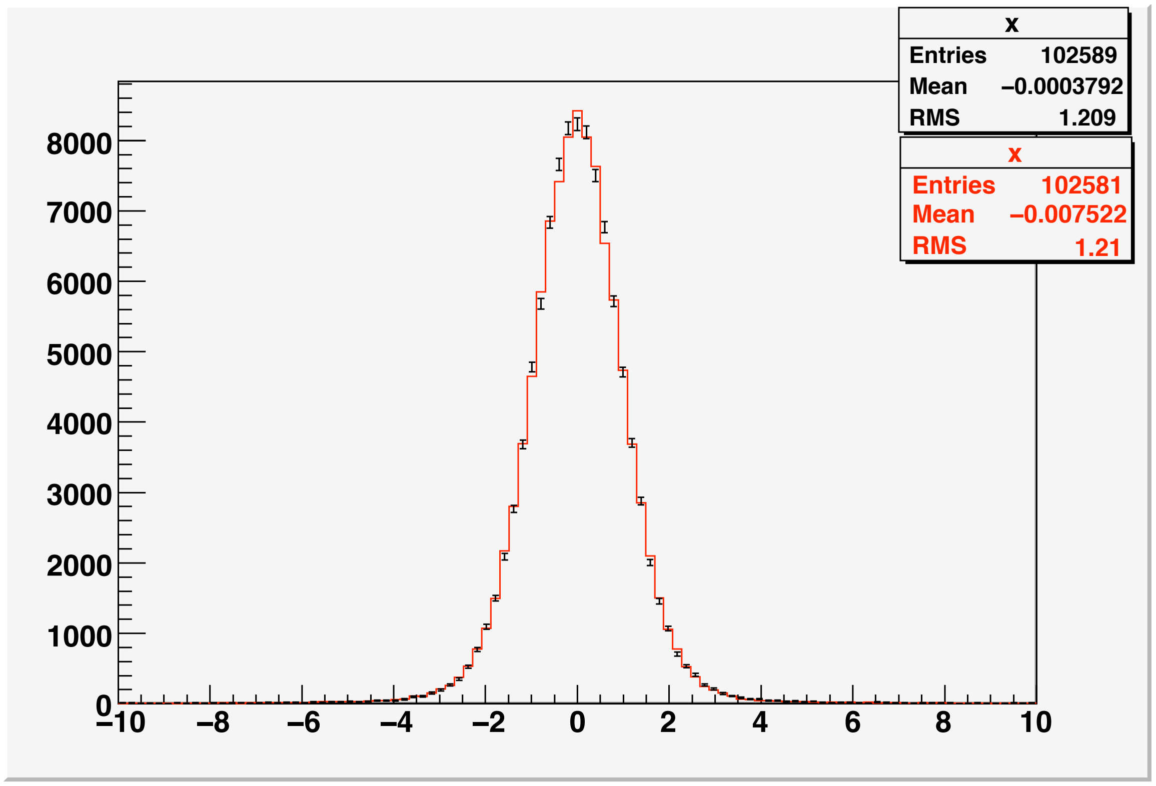
This differs from standard Geant4 tracking in the following ways:

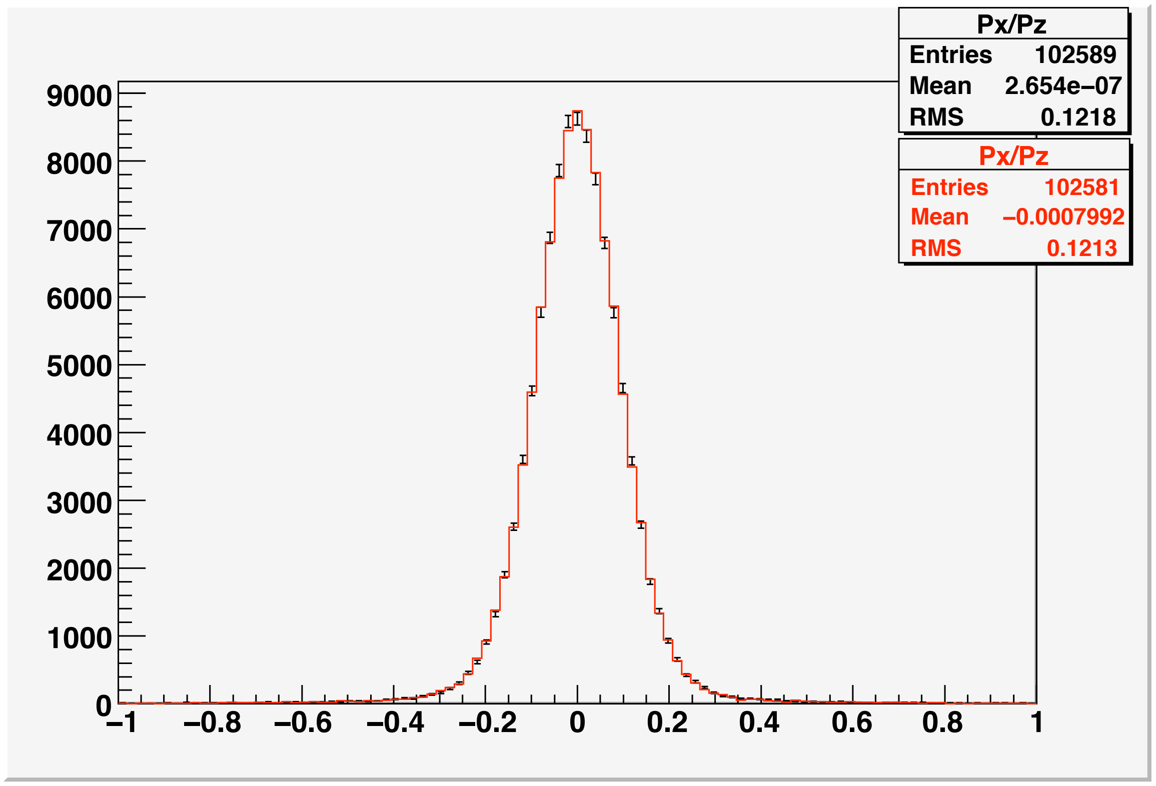
- There is a vector of tracks being tracked, not just one track.
- Each track has tracking completely re-started for each time step.
- As far as any Geant4 code is concerned, only a single track is active at any time; as far as the collectiveComputation() is concerned, all tracks are tracked in parallel, with their times equal to the value of the current time step.
- The collectiveComputation() function can be used to perform any sort of computation using the current vector of tracks; each command implements its specific computation, and multiple collective commands can be used in a single simulation. The computation can modify the tracks (e.g. by computing and applying momentum kicks).
- The use of random numbers is quite different from the standard tracking, so comparisons to standard tracking can be statistical only (not on an individual track-by-track basis).

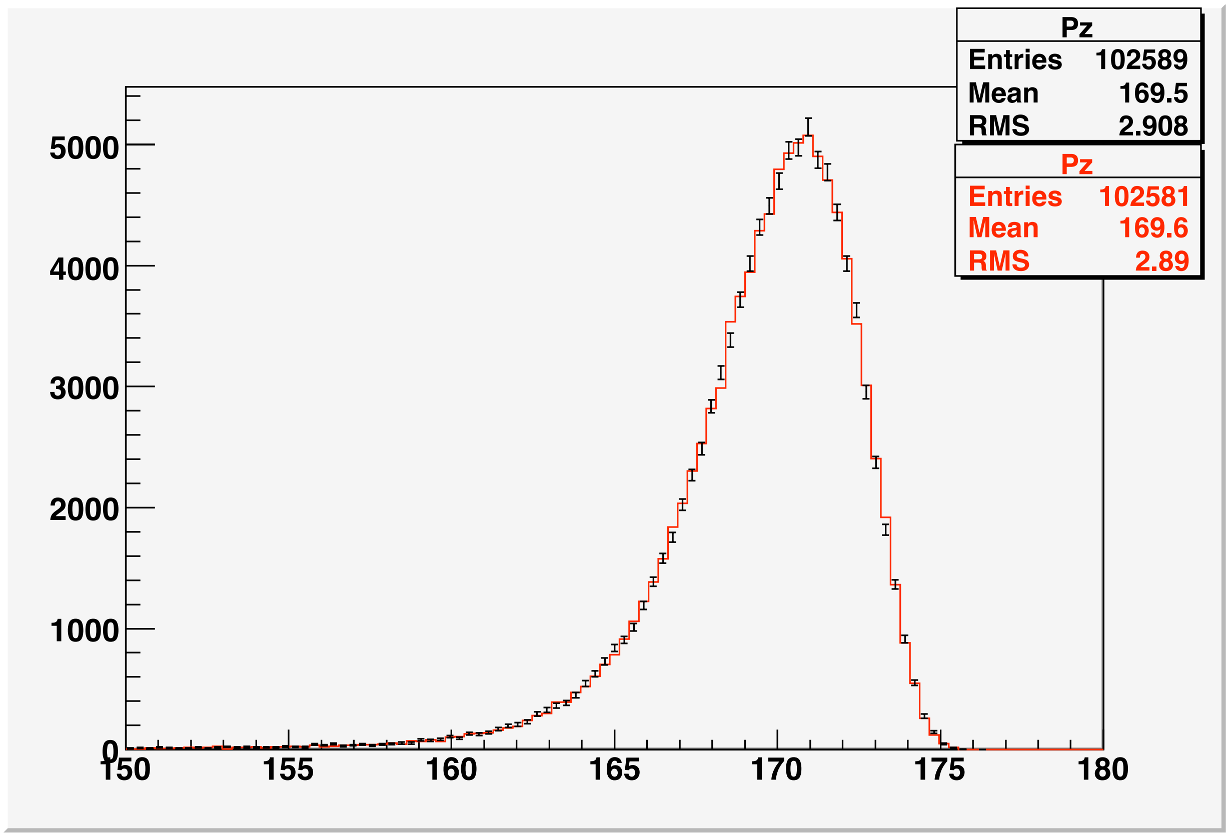
All Geant4 physics processes of interest tolerate this without problems. They of course apply only to each individual track.

Note that when using macroparticles to simulate larger bunches, each macroparticle is tracked as a single particle in the usual way, but their generated fields are multiplied by the macroparticle charge. It is non-trivial to determine what to do when one of the particles decays or interacts; it is easy to greatly increase the number of macroparticles, beyond the capabilities of memory or CPU. At present, the entire macroparticle follows the decay or interaction, which means the statistical accuracy can be poor (many more macroparticles are needed for accuracy).

The following plots compare standard tracking with collective tracking, using a collectiveComputation() that only monitors the tracks. The beam is 100,000 μ^+ with momentum 200 MeV/c and zero emittance, incident on a 20 mm slab of iron, followed by a virtualdetector to generate the plots. The collective time step is 0.01 ns. All secondaries and decay products are included in the plots. The standard tracking is plotted as black points with errorbars; the collective tracking is plotted in red. The standard run took 103 seconds; the collective run took 181 seconds.







The input file for this test is collective.g4bl:

```
*           collective.g4bl - test collective tracking
# comment or un-comment these lines
collective deltaT=0.01 verbose=1
param histoFile=collective
#param histoFile=standard
# keep individual steps shorter than the time step
param maxStep=1
physics QGSP_BERT
beam gaussian particle=mu+ meanMomentum=200 nEvents=100000
cylinder Target outerRadius=200 length=20 material=Fe
place Target z=1 front=1
virtualdetector Det radius=1000 length=1
place Det
```

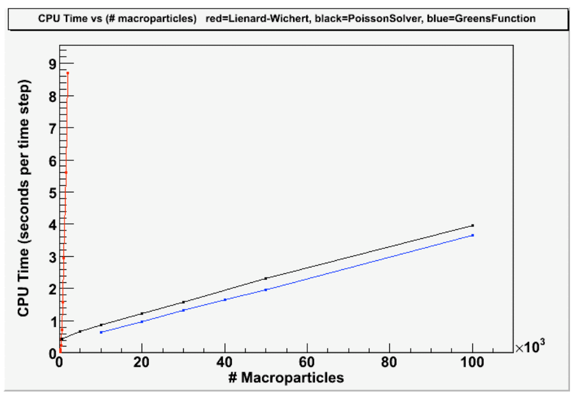
4.5 Space Charge

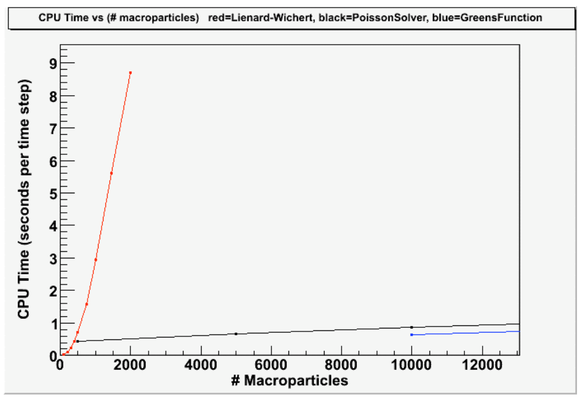
Three different implementations of the space-charge computation have been written and tested:

spacechargelw A computation based on Lienard-Wiechert potentials. This computation is CPU intensive, and in practice is limited to a few hundred macroparticles, which is too few for any realistic situation. But for the macroparticles used, it is accurate and correct. Its primary purpose is for testing the other computations.

spacechargeps	A computation using a solver for Poisson's equation on a 3-D grid in the beam frame. This computation requires a good approximation to the potential on the grid boundary, which can be difficult in practice. It relies on a Poisson solver written in Fortran, which limits its portability. As it has no benefits over the *spacecharge* command, and is much more finicky to use, it is not included in any G4beamline release, and is not discussed further.
spacecharge	A computation using the Green's function on a 3-D grid in the beam frame, based on convolution via FFTs. This is an excellent, efficient, and stable computation.

For the expansion of a Gaussian bunch of 10^{12} μ^+ expanding in free space, all three computations give the same result (to the width of the lines in the plots), but their CPU time requirements are very different, and they scale much differently with (# macroparticles), as shown here:





(Same plot, expanded horizontal scale.)

All three computations can handle multiple bunches of arbitrary particles, but spacechargeps and spacecharge require a reference particle for each bunch.

4.5.1 spacechargelw – Lienard-Wiechert computation

This computation uses the Lienard-Wichert formulas for E and B fields in the lab. These fields are simply included in the usual G4GlobalField (G4beamline uses a registration technique so many different elements and computations can generate fields, and overlaps are handled correctly and efficiently). Macroparticles are used to simulate much larger bunches than are feasible to track. Note that implementing this required using the collective computation discussed above.

The approximations involved are:

- Macroparticles are used to simulate large bunches.
- Particle trajectories are linearly interpolated between steps, and are occasionally extrapolated linearly for up to one time step from their last step.

- The radiation term is omitted.

This algorithm scales as (# macroparticles)^2, and is not computationally feasible for more than ~1,000 macroparticles. It was decided early on that the availability of a rigorously correct computation would permit the testing of other, more efficient approximations and algorithms (it is correct for the macroparticles used; whether that corresponds accurately to a real beam bunch is a different question, usually answered in the negative).

Test 1 – E and B fields from known sources
A 1 Coulomb charge has an electric field of 8.987551787E9 Volt/meter at a distance of 1 meter. This is tested by generating a single macroparticle with a charge of 1 Coulomb, at rest[1] at (x,y,z)=(0,0,0). At (x,y,z)=(1000,0,0), E_x=8988 MV/meter; at (x,y,z)=(0,2000,0), Ey=2247 MV/meter; at (x,y,z)=(0,0,-1000), E_z=-8988 MV/meter. These are the correct values and directions.

A uniform current of 1,000 Amps, at radius 0.1 m, has B=0.002 T. This is tested by generating 8,001 macroparticles[2] with charge +0.001 Coulomb, spaced uniformly 1 cm apart, moving at 10,000 meter/sec in the +z direction, centered at z=0. At (x,y,z)=(100,0,0), B_x= 0.00199999 Tesla; at (x,y,z)=(0,200,0), B_x=-0.000999988 Tesla. These are the correct values and directions to 5 significant digits.

There is no input file for these tests; they are executed automatically by the code whenever the spacechargelw command is used. There is an assert() to ensure they are correct.

Test 2 – Comparison to a textbook space-charge computation
The results are compared to a computation in the textbook by M. Reiser, Theory and Design of Charged Particle Beams, section 4.2.1, p200. This is a cylindrical bunch with uniform charge density; the comparison is at the longitudinal center of the bunch, where end effects are seen to be negligible.

The computation scales in many ways, and tests of proper scaling behavior were made for variations in:

- Macroparticle radius
- Macroparticle charge density exponent K
- Time step deltaT
- Total charge in the bunch
- Pz (longitudinal 3-momentum)
- R0 (initial radius of the bunch)
- Particle mass
- Sign of particle charge

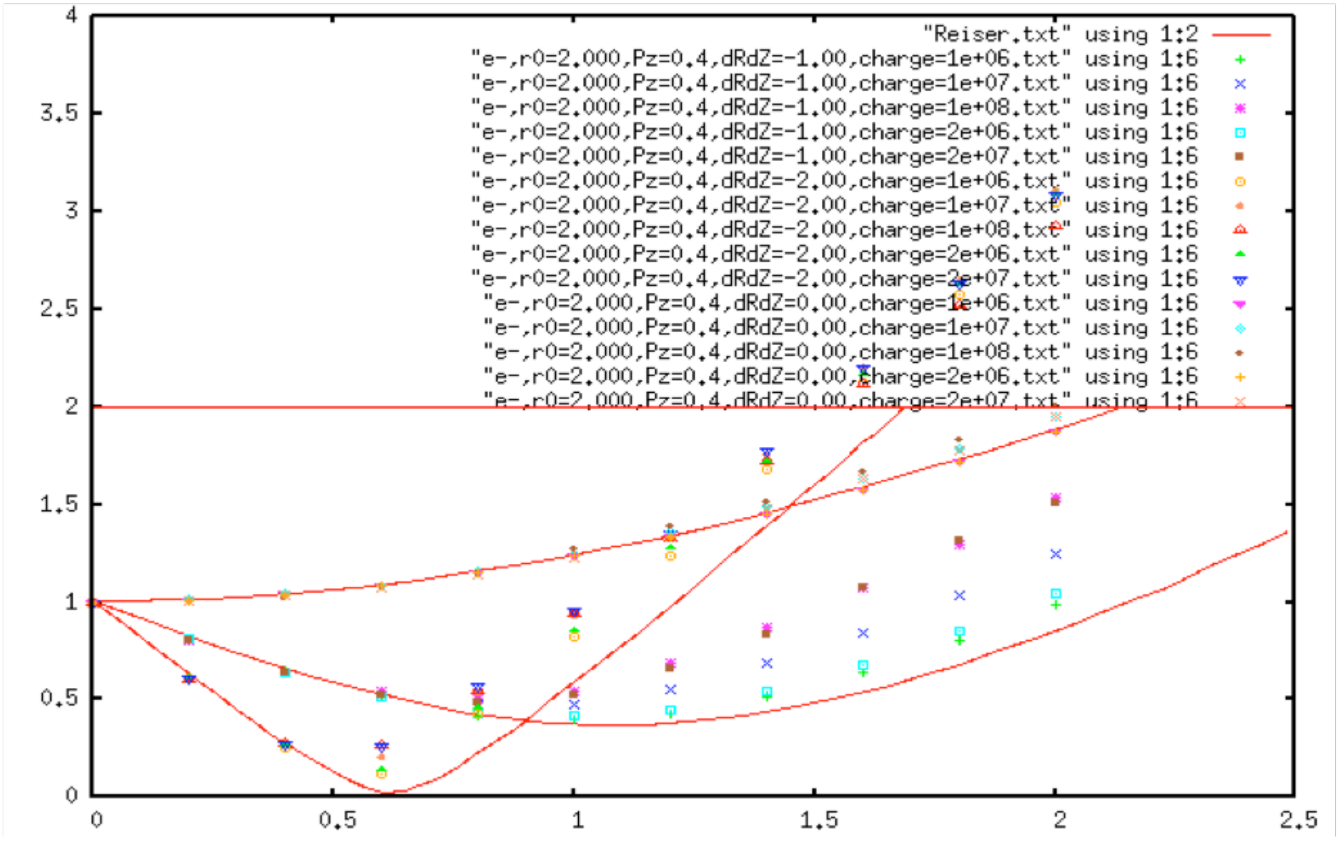
The scaling for number of macroparticles (total charge held constant) is violated in minor ways, as described in the next paragraph.

[1] This macroparticle is exactly at rest; its trajectory was generated manually, without any tracking.
[2] These macroparticle trajectories were also generated manually, without any tracking.

In general, the algorithm gives more expansion than Reiser's computation. This appears to be due to the finite number of macroparticles – using more macroparticles with less charge in each gives results closer to the computation. The plot below keeps the total charge of the bunch constant, but varies the number of macroparticles from 100 to 10,000. The vertical axis is the scaled radius of the bunch and the horizontal axis is the scaled position along z of the bunch center; both axes are scaled by factors involving Pz, particle mass, initial radius, and total charge -- see the textbook for details. The three red lines correspond to three initial conditions:

dR/dZ=0	momenta parallel to the Z axis
dR/dZ=-1	focused to a point at Z=1
dR/dZ=-2	focused to a point at Z=0.5

The first case is pretty good for just 100 macroparticles. The second case is not too bad for 10,000 macroparticles. The third case clearly needs significantly more than 10,000 macro particles for good accuracy. Unfortunately, this algorithm scales as (# macroparticles)^2, and it becomes computationally infeasible to use more macroparticles[1].



4.5.2 spacecharge – Beam-frame Green's Function Convolution via FFTs

This computation inherently requires the beam to be bunched. Macroparticles are used to simulate much larger bunches than are feasible to track. It requires that all particles of a bunch be near to its reference particle, in both position and 3-momentum. The particles of the bunch are boosted to the rest frame of the reference, and a 3-d rectangular grid is defined that contains the

[1] The run with 10,000 macroparticles took 14 hours of CPU time, for a mere 400 mm of travel.

particles. The particles are placed into this grid, and Poisson's equation is solved on the grid for the electrostatic potential in the beam frame. This solution is obtained by convolving the Green's function (infinite boundary conditions) with the source distribution; FFTs are used for efficiency. This is differentiated to obtain the E field, which is then boosted back to the lab to determine the E and B fields for tracking. This is performed at each time step to handle the evolution of the particles within the bunch. The grid moves with the reference particle between time steps, and is dynamically sized to keep the 99th percentile of the charge distribution between 0.5 and 0.67 of the grid size (separately in x, y, and z).

FFTW 3.2.2 [11] is used to implement the fast Fourier transforms. The beam-frame grid is doubled in all three dimensions so the cyclical convolution via FFTs yields the correct potential for infinite boundary conditions [12]. As the Green's function includes the boundary conditions, the difficulties of *spacechargeps* are avoided.

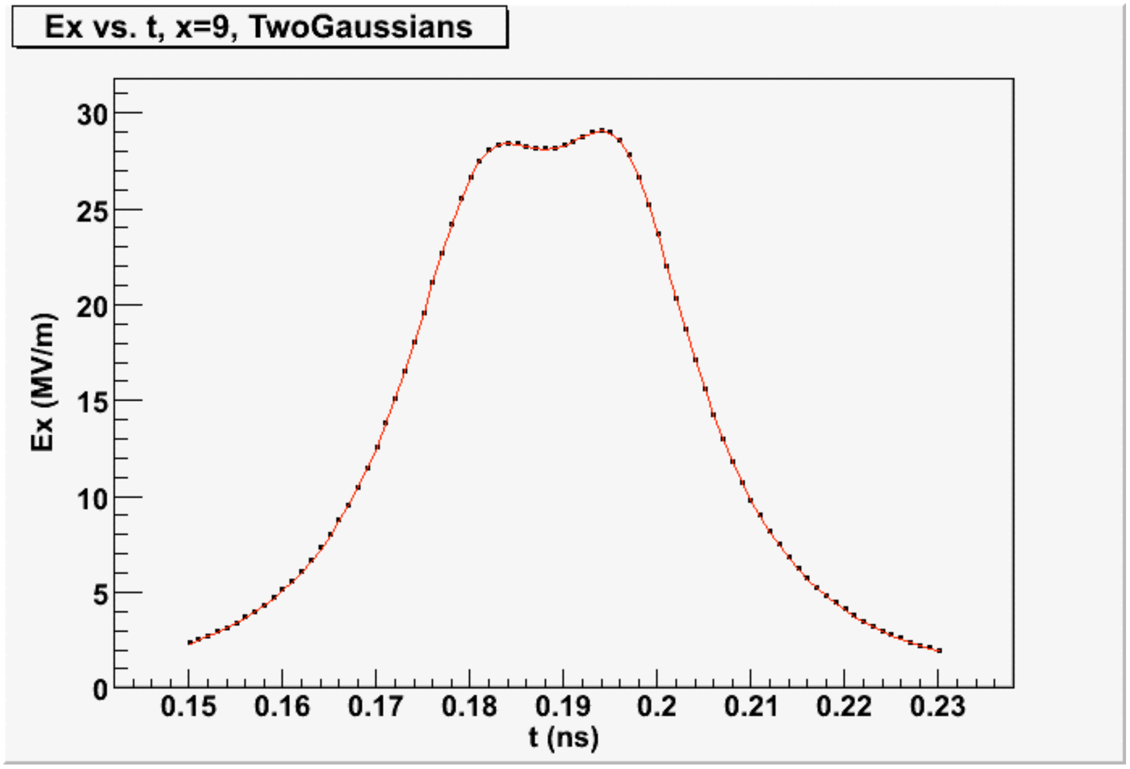
Test 1 – Comparison to known sources
A 1 Coulomb charge has an electric field of 8.987551787E9 Volt/meter at a distance of 1 meter. This is tested by generating a single macroparticle with a charge of 1 Coulomb, at rest at (x,y,z)=(0,0,0). At (x,y,z)=(1000,0,0), E_x=8987.54 MV/meter.

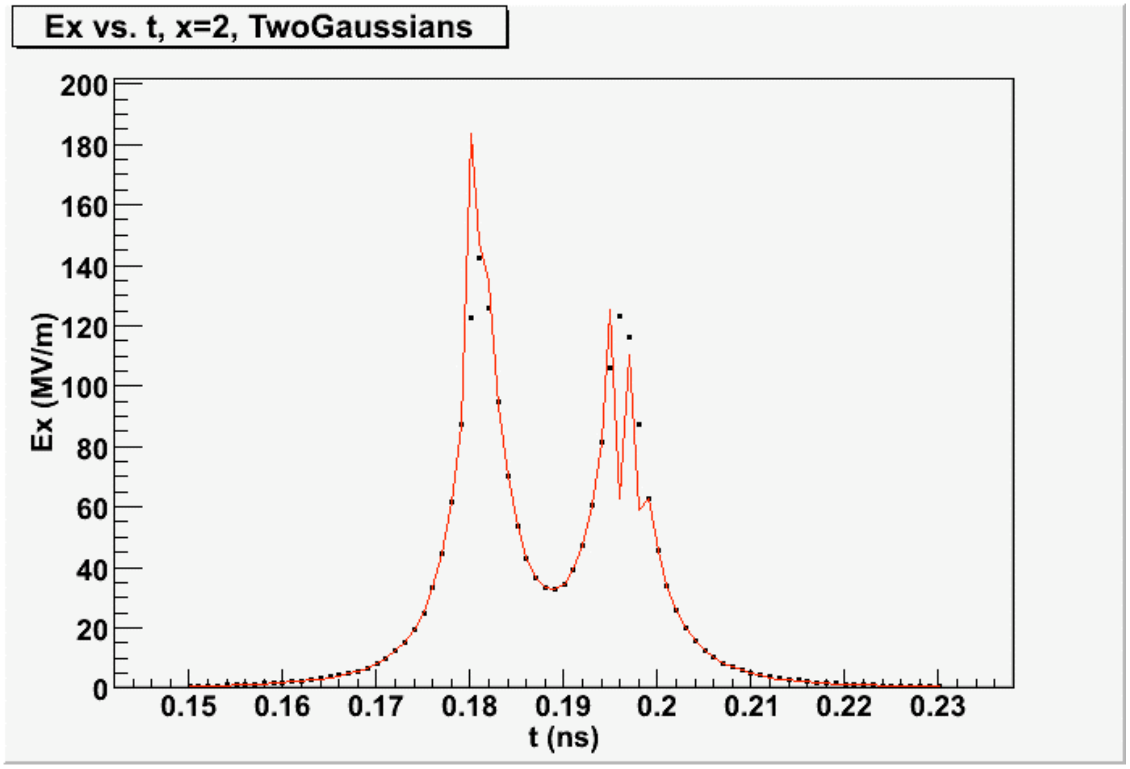
A uniform current of 1,000 Amps, at radius 0.1 m, has B=0.002 T. This is tested by generating 501 macroparticles with charge +0.001 Coulomb, spaced uniformly 1 cm apart, moving at 10,000 meter/sec in the +z direction, centered at z=0. At (x,y,z)=(100,0,0), B_x= 0.001971 Tesla.

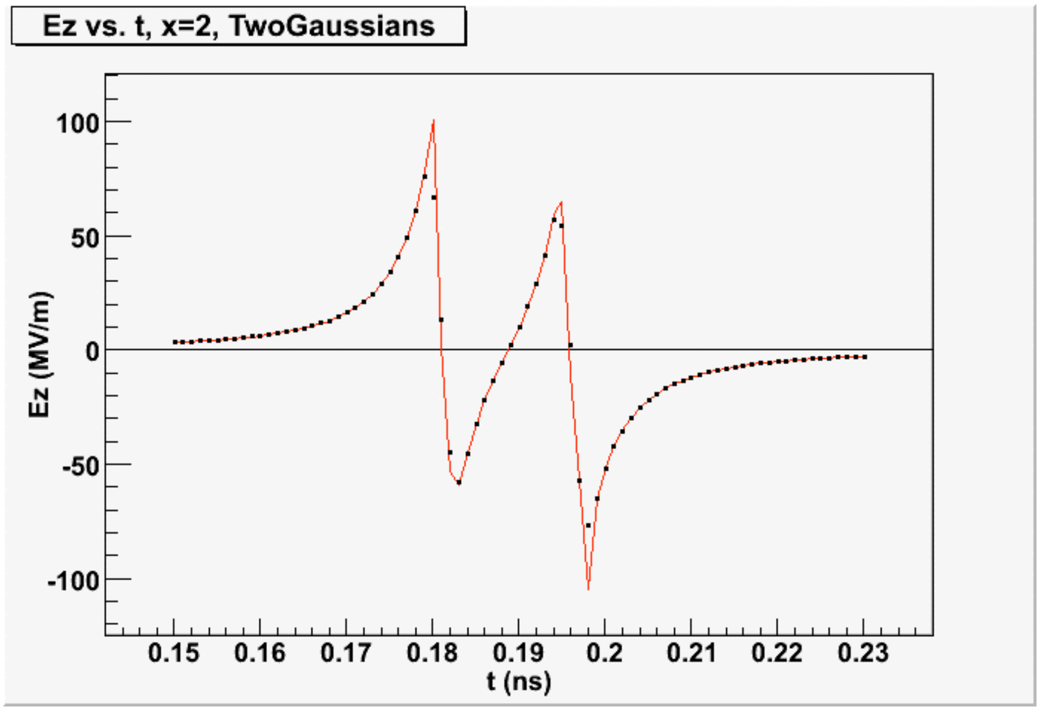
There is no input file for these tests; they are executed automatically by the code whenever the spacecharge command is used. There is an assert() to ensure they are correct.

Test 2 – Comparison to the Lienard-Wiechert computation
The ability to compare two different calculations is a powerful capability. This code and its algorithm are so completely different from those in spacechargelw that the agreement shown here gives confidence in the correctness of both. Note, however, that it shares considerable code in common with spacechargeps (everything but the actual solving of Poisson's equation on the grid).

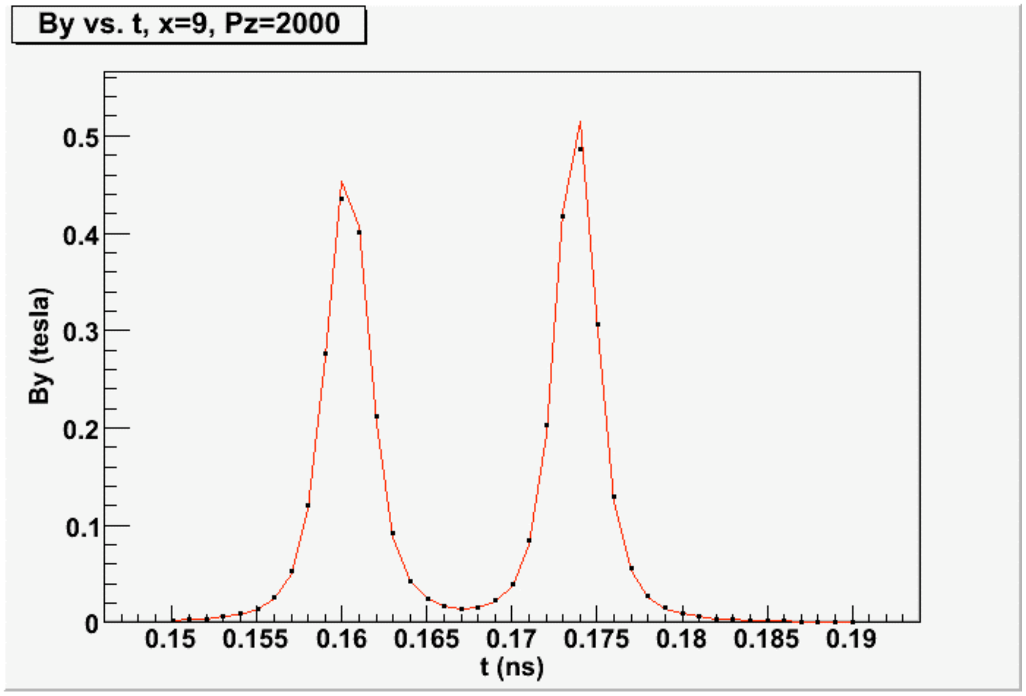
Here are plots of the E field at a fixed point in space as a function of time, during the transit of a bunch consisting of two Gaussians. The total bunch is 10^{12} μ^+ with Pz=200 MeV/c, consisting of two Gaussian distributions in space with σ_x=σ_y=2mm, σ_z=0.2mm, separated by 4mm along z. The spacechargelw computation is a red line, and the spacecharge computation is black dots; they use exactly the same set of 2,000 macroparticles. It is clear that some spatial resolution is lost, due to the spatial averaging inherent in a grid, but that the computation is correct within the approximation that defines its limitations. x=2 is well inside the bunch and the grid; x=9 is outside the bunch but inside the grid.


Ex vs. t, x=9, TwoGaussians
Ex (MV/m)
t (ns)
0
5
10
15
20
25
30
0.15
0.16
0.17
0.18
0.19
0.2
0.21
0.22
0.23

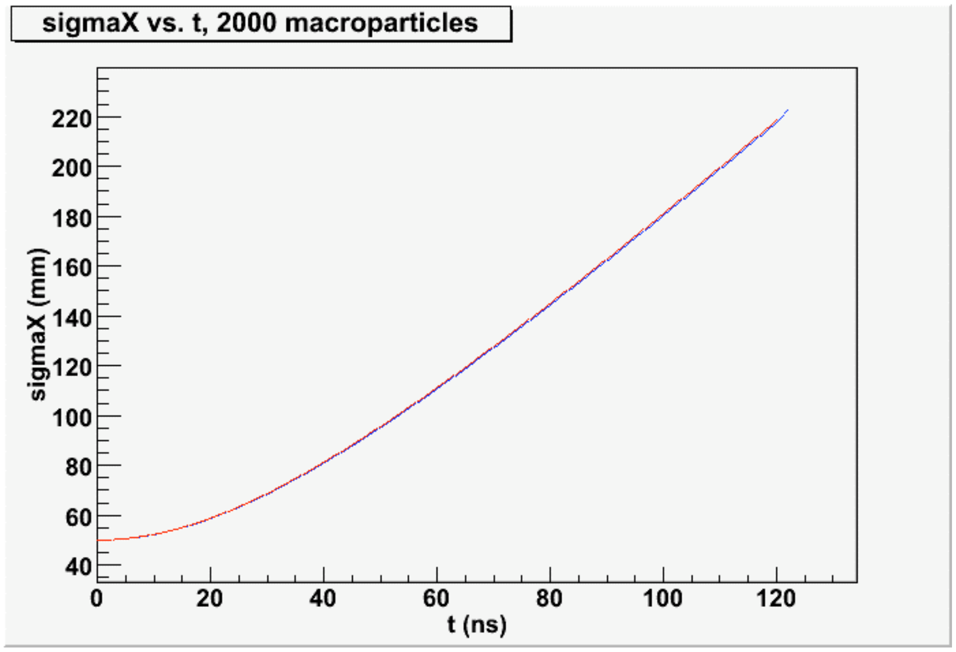

Ex vs. t, x=2, TwoGaussians
Ex (MV/m)
t (ns)
0
20
40
60
80
100
120
140
160
180
200
0.15
0.16
0.17
0.18
0.19
0.2
0.21
0.22
0.23



Here is a similar plot, showing B_y vs. t at x=9, but for a beam momentum of 2,000 MeV/c. The higher momentum increases the value of B, and reduces the time it takes to reach the point at which the fields are sampled; because the particles are now much more relativistic (γ=19), the widths of the Gaussians are reduced so they are well resolved (compared to E_x at x=9 above). Given the G4beamline coordinates with x=beam left, y=up, z=beam, and positive particles, the sign is correct.



Another comparison is to plot the expansion of a Gaussian bunch with initial parameters $\sigma_x = \sigma_y = \sigma_z = 50$ mm, $\sigma_{x'} = \sigma_{y'} = \sigma_{Pz} = \sigma_t = 0$; the bunch consists of 10^{12} μ^+, each particle has a kinetic energy of 100 keV, propagating along Z in free space. The plot is sigmaX; sigmaY and sigmaZ behave identically. The plot contains 2,000 macroparticles. The KE is so low that this corresponds accurately to a bunch at rest in the lab.



Test 3 – Comparison to a textbook space-charge computation
The results are compared to a computation in the textbook by M. Reiser, Theory and Design of Charged Particle Beams, section 4.2.1, p200. This is a cylindrical bunch with uniform charge density; the comparison is at the longitudinal center of the bunch, where end effects are seen to be negligible.

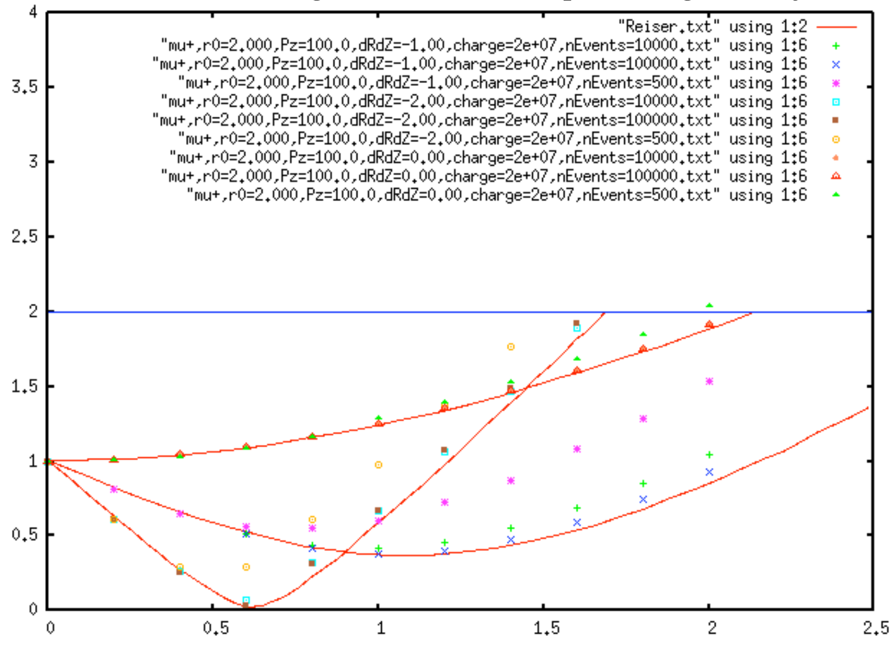
The computation scales in many ways, and tests of proper scaling behavior were made for variations in:

- Time step deltaT
- Total charge in the bunch
- Pz (longitudinal 3-momentum)
- R0 (initial radius of the bunch)
- Particle mass
- Sign of particle charge

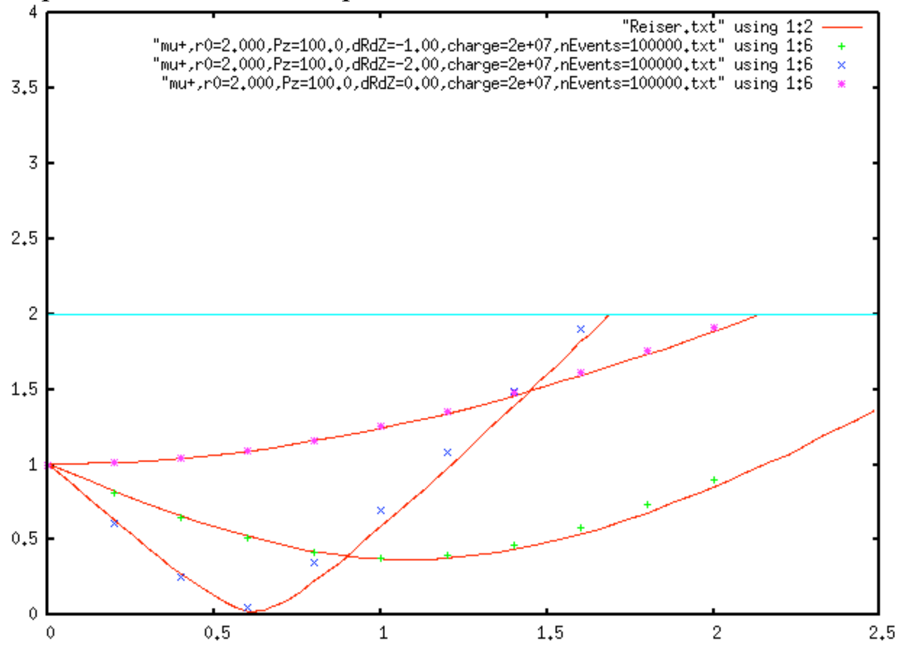
The plot below keeps the total charge of the bunch constant, but varies the number of macroparticles from 2,000 to 100,000. The grid is 65x65x513; its physical size is kept between 3/2 and 2 times the 99th percentile of the bunch dimensions. The vertical axis is the scaled radius of the bunch and the horizontal axis is the scaled position along z of the bunch center; both axes are scaled by factors involving Pz, particle mass, initial radius, and total charge – see the textbook for details. The three red lines correspond to three initial conditions:

dR/dZ=0	momenta parallel to the Z axis
dR/dZ=-1	focused to a point at Z=1
dR/dZ=-2	focused to a point at Z=0.5

This plot is the best of all, and shows the advantage of being able to efficiently track 100,000 macroparticles. It also shows that using a mere 500 macroparticles gives major inaccuracies.



Reducing the grid to 33x33x257 (factor of 8 fewer grid points) does not affect the agreement very much. This plot has 100,000 macroparticles.



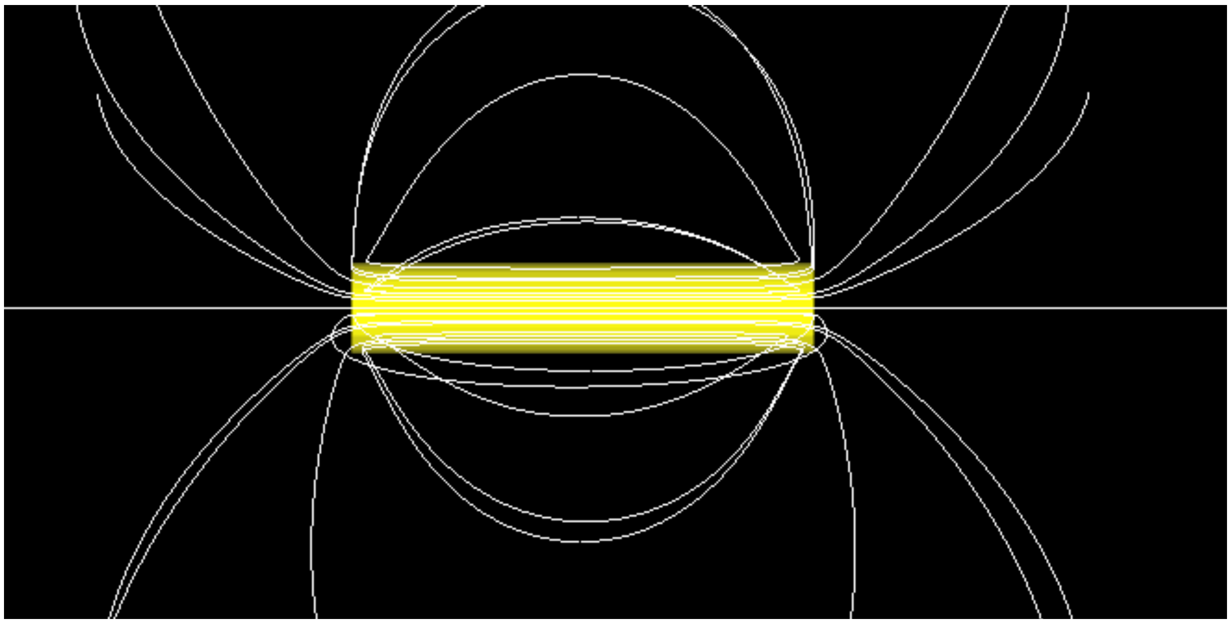
4.6 Neutrino Interactions

Neutrino interactions are not yet implemented in G4beamline. It is expected they will be implemented using an external program to model the generation of secondary particles, which will then be tracked as usual by G4beamline. The interaction cross-section will be greatly enhanced.

5 Beamline Elements Implemented in G4beamline

5.1 coil and solenoid

The field lines look reasonable (these are in 3-D, and apparent intersections are not real; to really appreciate this you must run it and manipulate the image in real time to look at it from different directions.



The input file for this plot is solenoid.g4bl:

```
*	solenoid.g4bl - solenoid field lines
physics QGSP_BERT
coil C innerRadius=199.5 outerRadius=200.5 length=2000
solenoid S coil=C current=1000 color=1,1,0,.5
place S z=0
cylinder K outerRadius=20000 length=1 color=''
place K z=10000
printfield field=Bz layout=zx drow=50 nrow=100 dcol=1 ncol=1 x=0 y=0 z=0
fieldlines center=0,0,0 radius=200 dl=1 exit=1 nLines=10
```

Test 2 – Helmholtz Coils field value

From Reitz and Milford, Foundations of Electromagnetic Theory, p158, the field at the midpoint of a pair of Helmholtz coils is

$$B_z = 32\,\pi\,N\,I / (5^{3/2}\,a\,10)$$

where I is the current in amperes, a is the coil radius in cm, N is the number of turns, and B_z is given in gauss. The coil separation is of course equal to their radius. For N=1, I=10000 A, a=50 cm, this formula gives 179.84 gauss. G4beamline gives 0.017984 tesla, which is the same.

The input file for this test is helmholtz.g4bl:

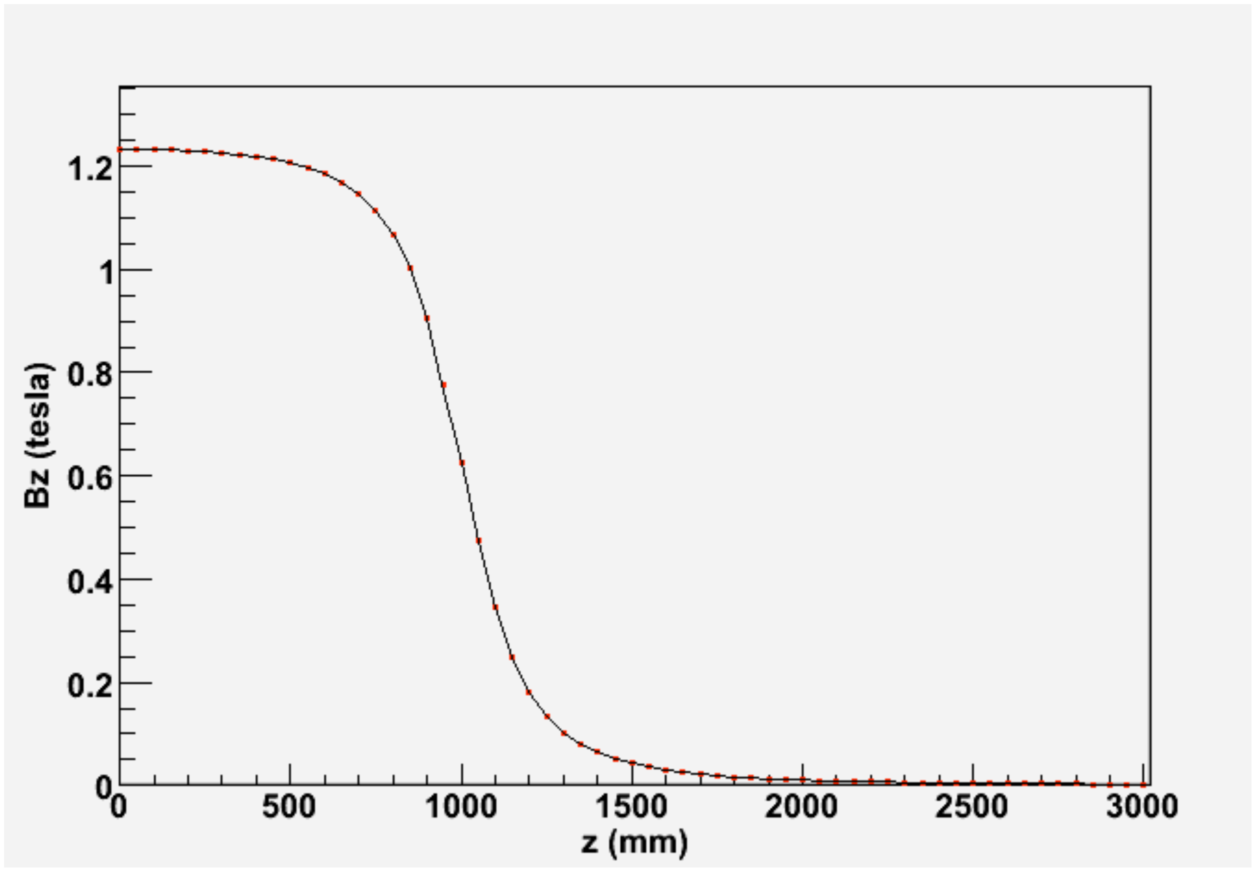
```
*	helmholtz.g4bl - test Helmholtz coils
physics QGSP_BERT
coil C innerRadius=499.5 outerRadius=500.5 length=1 maxR=500 maxZ=500
# coil C has an area of 1 mm^2, so current in amp/mm^2 is also amps.
solenoid S coil=C current=10000
place S z=-250
place S z=250
probefield
```

Test 3 – On-Axis Field

From R. Winch, Electricity and Magnetism, p380, on the axis of a solenoid the B field is:

$$B_z = \mu_0\,N\,I\,(\cos\alpha - \cos\beta) / 2\,L$$

Where μ_0 is $4\pi\times10^{-7}$, N is the number of turns, I is the current in amps, L is the solenoid length in meters, and α and β are angles from the point on the axis to the ends of the solenoid; B_z is given in tesla. The following plot compares this formula to G4beamline, for a solenoid of radius 200 mm and a length of 2,000 mm, centered at z=0. The formula is a black line and the G4beamline data are red points:



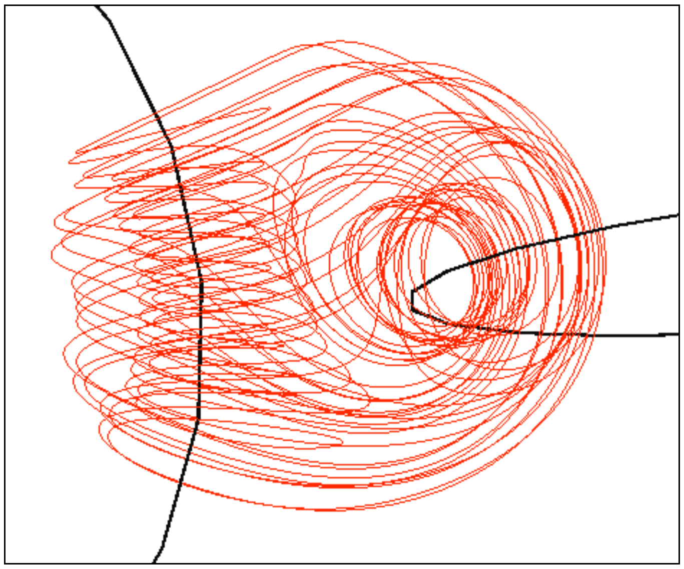
G4beamline creates a field map for the solenoid, in this case using the default tolerance of 0.1% to determine the grid spacing and extent of the map. Numerically, the two data series above differ by at most 0.0004 tesla (0.03%) at every point within the map (which extends beyond the plot to z=3755 mm), and at most by 0.0007 tesla (0.06%) outside the map. The points plotted are not related to the field map's grid spacing.

The input file for this test is solenoid.g4bl:

```
*	solenoid.g4bl - solenoid field lines
physics QGSP_BERT
coil C innerRadius=199.5 outerRadius=200.5 length=2000
solenoid S coil=C current=1000 color=1,1,0,.5
place S z=0
cylinder K outerRadius=20000 length=1 color=''
place K z=10000
printfield field=Bz layout=zx drow=50 nrow=100 dcol=1 ncol=1 x=0 y=0 z=0
fieldlines center=0,0,0 radius=200 dl=1 exit=1 nLines=10
```

Test 4 – Chaotic B field

The CERN Courier [13] showed a plot of a chaotic B field from two circular current loops at right angles to each other. The plot of a single field line below is quite similar to the picture they showed. Indeed, moving the initial position of the field line by 0.1 mm changes its character completely. This is not due to the approximation introduced by the field map of the coil – using the exact computation shows similar behavior (but different in detail, as is expected for such chaotic behavior).

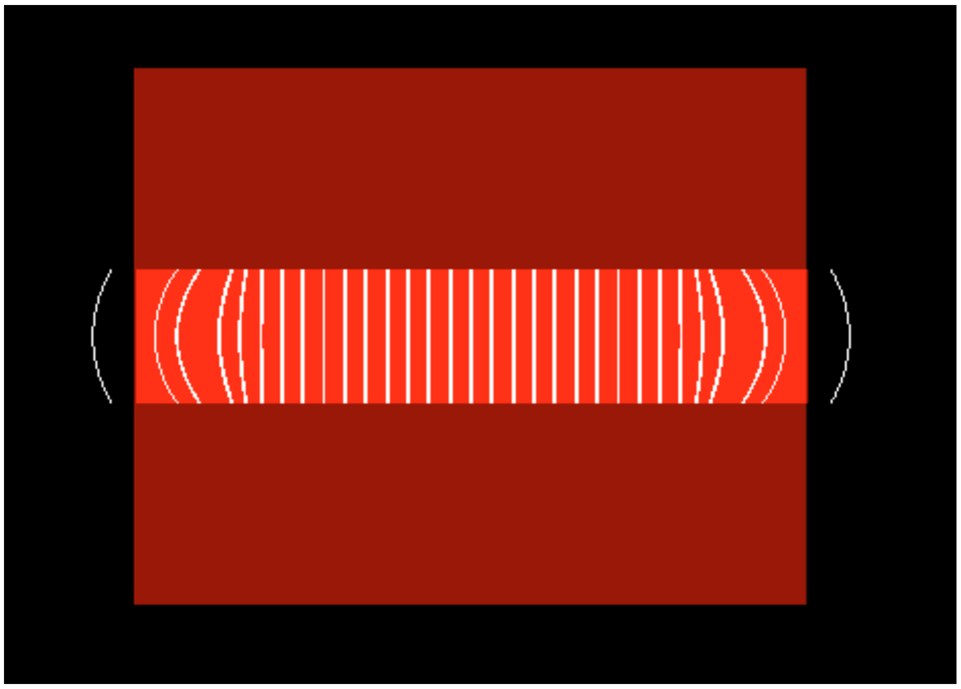


The input file for this test is chaos.g4bl:

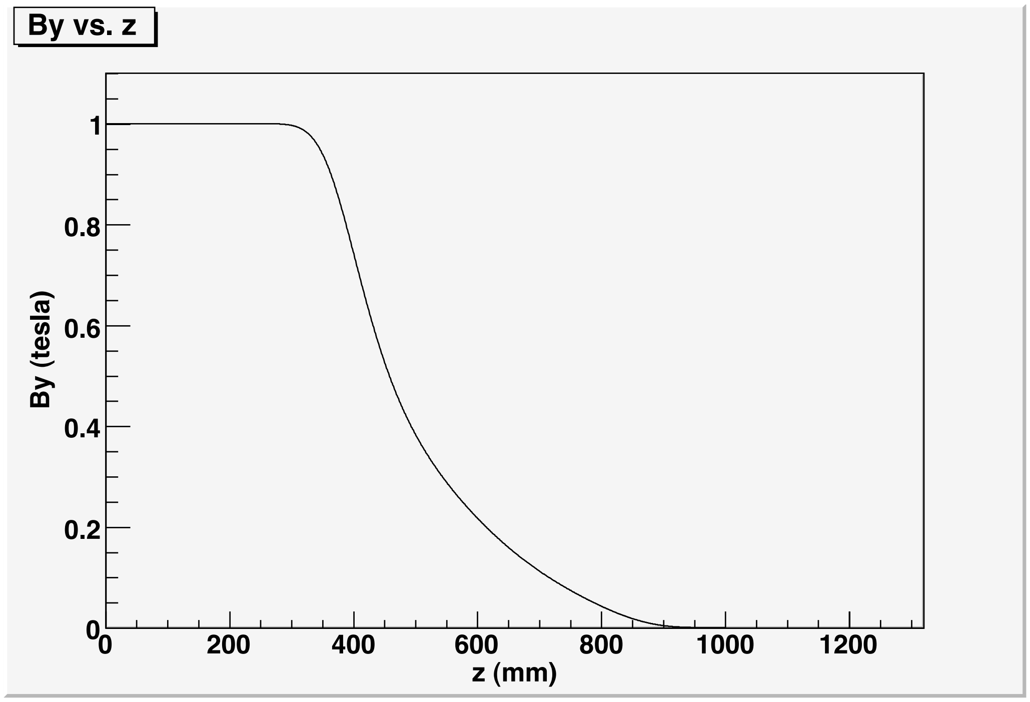
```
*	chaos.g4bl
physics QGSP_BERT
coil C innerRadius=199.5 outerRadius=200.5 length=1 nSheets=1 \
    exactComputation=1
solenoid S coil=C current=10000 color=0,0,0
box Huge height=50000 width=50000 length=50000 color=invisible
place S z=0 x=250 parent=Huge
place S z=0 x=-250 parent=Huge rotation=Y180,X90
place Huge z=0
g4ui when=4 "/vis/viewer/set/background 1 1 1"
fieldlines exit=1 nLines=0 forever=1 dl=1 67.1,0,0 color=1,0,0
```

5.2 genericbend

This plot of field lines shows how artificial the region containing the field is – the field is valid only within the aperture, and within the aperture extended outside. In particular, the field is zero inside the iron. The fringe field does not include effects from the sides, only the top and bottom. The field lines are in 3-D, and this side view does not accurately reflect their density; it does show the behavior of the fringe fields. The field lines look reasonable; to really appreciate this you must run it and manipulate the image in real time to look at it from different directions.



The following plot shows B_y along the z axis, as seen by the ν_μ beam particle trace.



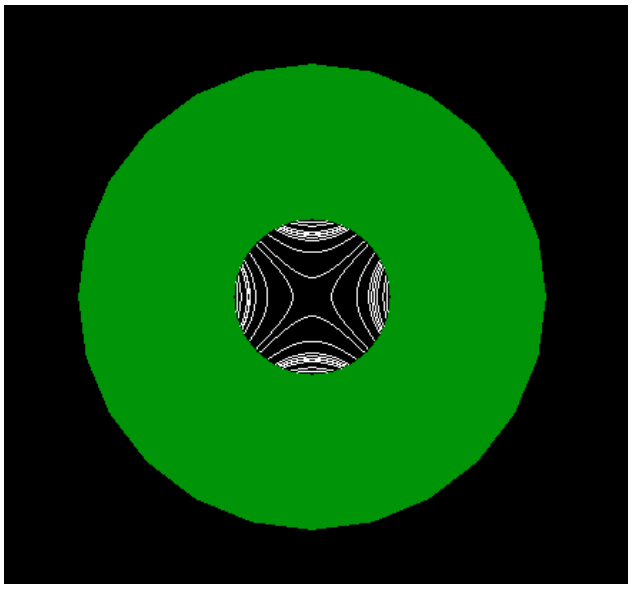
The sum of the B_y entries in the trace file is 500.501, corresponding approximately to the field integral $\int B_y dl$ in tesla-mm. The correct value is 500.000, so the fringe-field computation has indeed preserved the field integral, to within the accuracy of this simple integration technique.

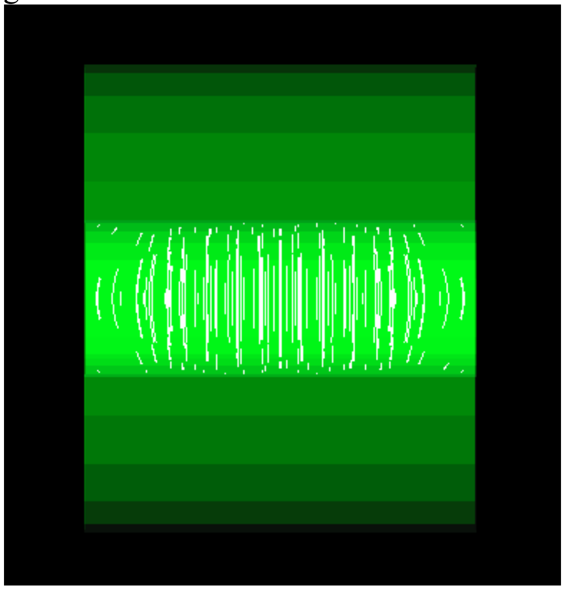
The input file for these two plots is bend.g4bl:

```
*         bend.g4bl genericbend field lines
physics QGSP_BERT
beam gaussian particle=nu_mu nEvents=1
trace nTrace=1 format=ascii
param maxStep=1
genericbend B1 fieldWidth=500 fieldHeight=200 fieldLength=1000 \
   ironWidth=1000 ironHeight=800 ironLength=1000 By=1 openAperture=1
ironColor=1,0,0,.5
place B1 z=0
fieldlines center=0,0,0 radius=2000 square=1 dl=1 nLines=400 exit=0
fieldexpr F height=200 width=1000 length=2000 By=0.001001
place F z=0
box end height=1 width=1 length=1
place end z=2000
```

5.3 genericquad

This plot of field lines shows how artificial the region containing the field is – the field is valid only within the aperture, and within the aperture extended outside. In particular, the field is zero inside the iron. The field lines are in 3-D, and these views do not accurately reflect their density, but the side view does show the behavior of the fringe fields.





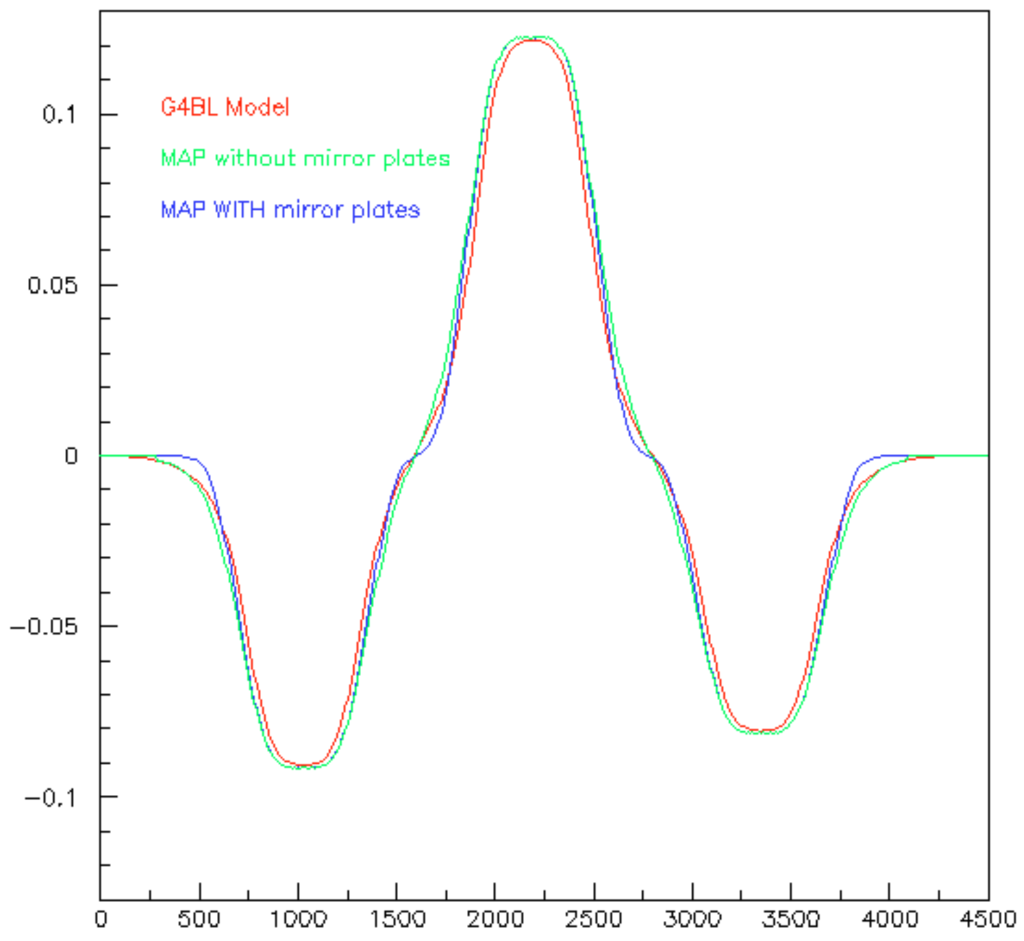
The input file for this test is quad.g4bl:

```
*         quad.g4bl genericbend field lines
physics QGSP_BERT
genericquad Q1 fieldLength=1000 ironLength=1000 apertureRadius=200 \
   ironRadius=600 gradient=4 ironColor=0,1,0,0.5
place Q1 z=0
fieldlines center=50,0,0 radius=200 square=1 dl=1 nLines=10 exit=0
fieldlines center=-50,0,0 radius=200 square=1 dl=1 nLines=10 exit=0
fieldlines center=0,-50,0 radius=200 square=1 dl=1 nLines=10 exit=0
fieldlines center=0,50,0 radius=200 square=1 dl=1 nLines=10 exit=1
```

Test 2

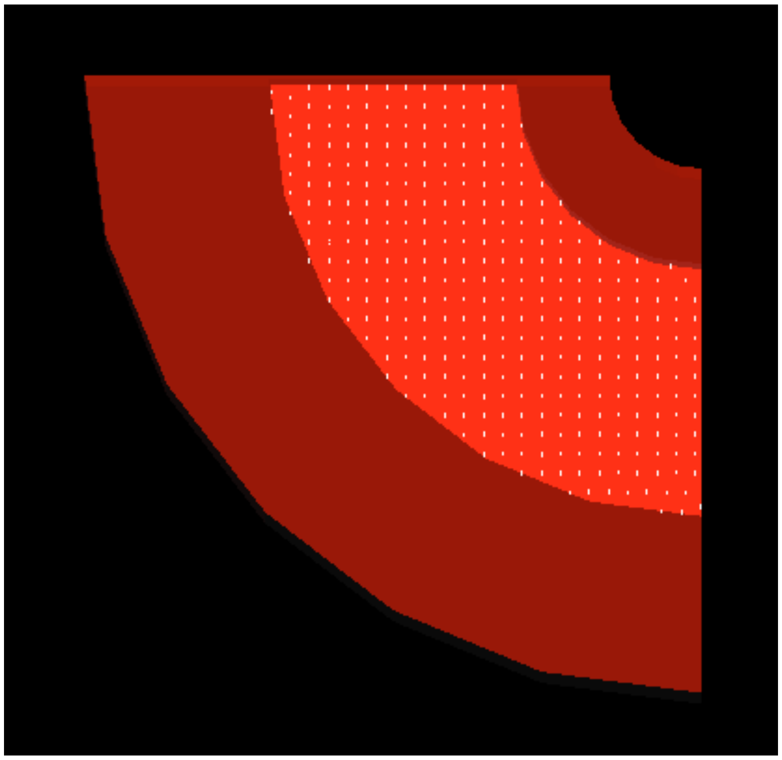
Marco Apollonio has compared the G4beamline genericquad field to that of three TypeQC quadrupole magnets in a row. The following plot compares field maps generated by Tosca to the genericquad field. The mirror plates are 1-inch iron plates at the front and back to "reflect" the field and thus reduce the extent of the fringe field along the axis; genericquad has no mirror plates. The Tosca-generated maps were validated against measurements of one magnet, but the details have been lost. This is a large quadrupole magnet with a pole-tip radius of 171 mm and an overall length of 1,046 mm; its large diameter-to-length ratio implies that fringe fields are important. Drawings of the magnet are on pages 11-12 of http://hep04.phys.iit.edu/cooldemo/micenotes/public/pdf/MICE0065/MICE0065.pdf.

The following plot is of B_y along a line parallel to the z axis but off axis in x.



5.4 idealsectorbend

This plot of field lines shows how artificial the region containing the field is – the field is valid only within the aperture. In particular, the field is zero inside the iron. This element has no fringe field. The field lines look reasonable; to really appreciate this you must run it and manipulate the image in real time to look at it from different directions.



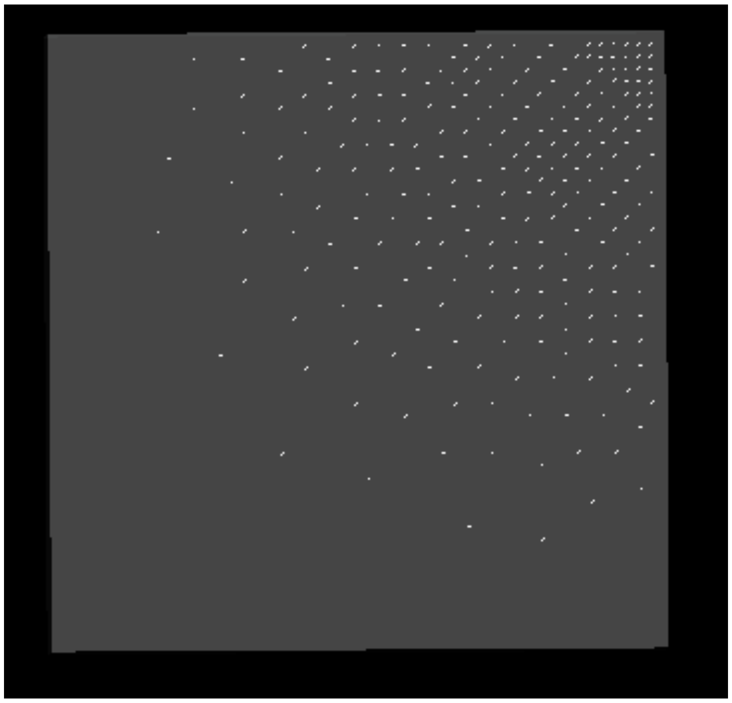
(Some field lines appear to be in the edge of the iron; they are not. This picture is rotated slightly to ensure the field lines are visible.)

The input file for this test is sectorbend.g4bl:

```
*       sectorbend.g4bl idealsectorbend field lines
physics QGSP_BERT
idealsectorbend B1 angle=90 fieldCenterRadius=500 fieldInnerRadius=300  \
    fieldOuterRadius=700 fieldHeight=400 ironInnerRadius=150 \
    ironOuterRadius=1000 ironHeight=800 ironColor=1,0,0,.5 By=1
place B1 z=0
fieldlines center=146,0,353 radius=500 square=1 dl=1 nLines=500 exit=1
```

5.5 fieldmap

This plot of field lines shows a fieldmap with Bz proportional to x*y, looking from the z axis; the field is valid only in a 1-meter cube, shown in dark gray. The field lines look reasonable; to really appreciate this you must run it and manipulate the image in real time to look at it from different directions.

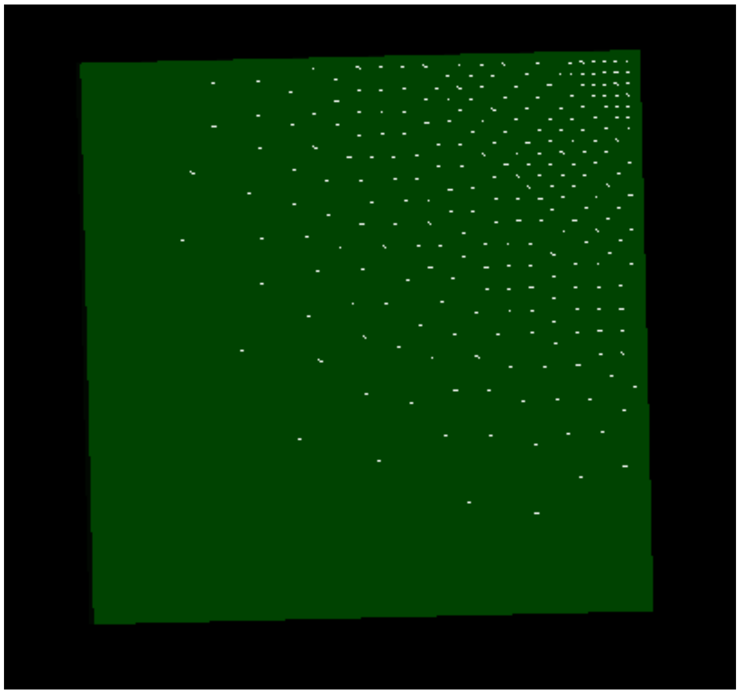

(The image is rotated slightly to ensure the field lines are visible. x=0,y=0 is in the lower left corner.)

The input file for this test is fieldmap.g4bl:

```
#	fieldmap.g4bl - test fieldmap
physics QGSP_BERT
fieldmap Map file=fieldmap.B
place Map z=0
box Vis height=1000 width=1000 length=1000 color=1,1,1,0.2
place Vis x=500 y=500 z=500
fieldlines center=500,500,500 radius=500 square=1 exit=1 nLines=500 N=50
```

5.6 fieldexpr

This plot of field lines shows a fieldmap with Bz proportional to x*y, looking from the z-axis; the field is valid only in a 1-meter cube, shown in dark green. The field lines look reasonable; to really appreciate this you must run it and manipulate the image in real time to look at it from different directions.

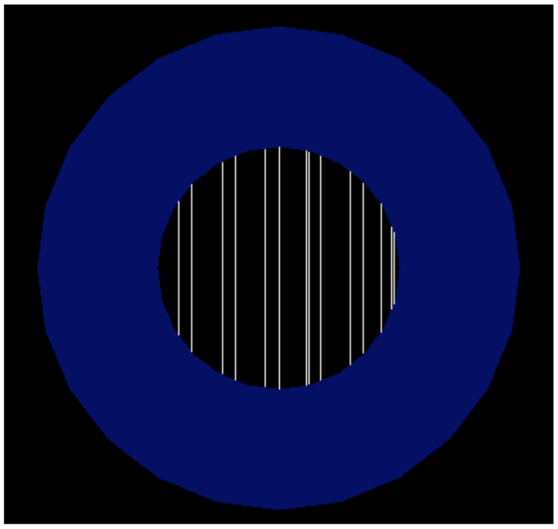
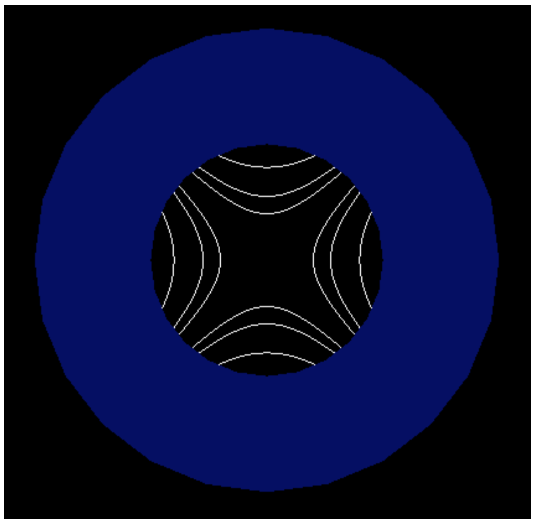
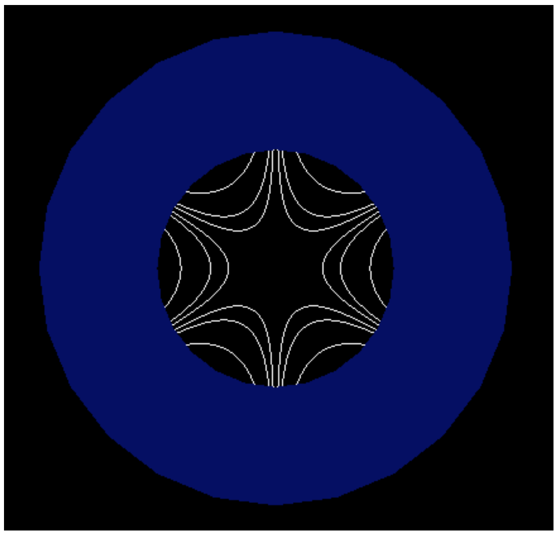
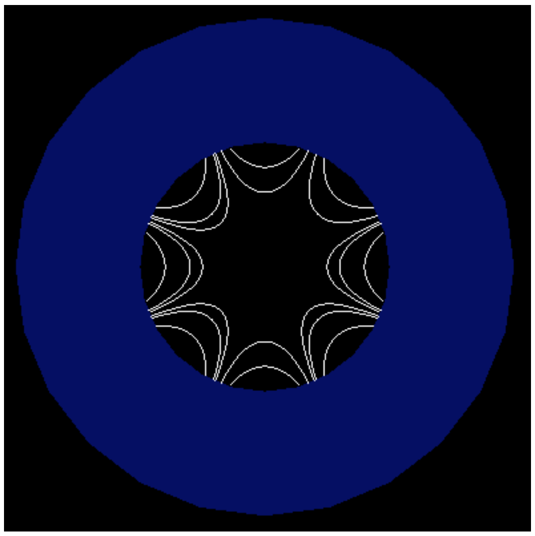
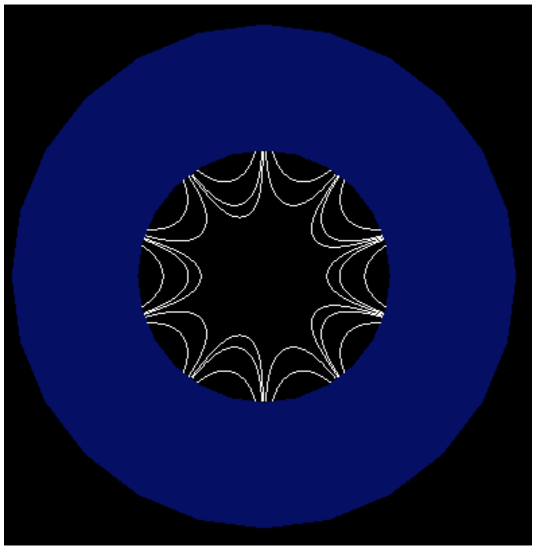
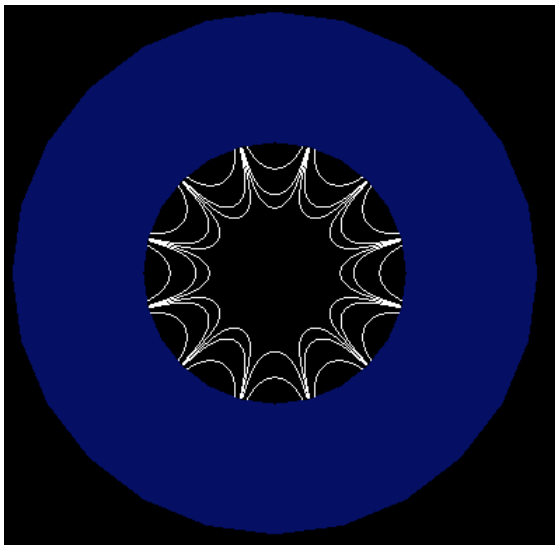

(The image is rotated slightly to ensure the field lines are visible. x=0,y=0 is in the lower left corner.)

The input file for this test is fieldexpr.g4bl:

```
#	fieldexpr.g4bl - test fieldexpr
physics QGSP_BERT
fieldexpr Expr height=1000 width=1000 length=1000 Bz=(x+500)*(y+500)/1000
place Expr z=0
box Vis height=1000 width=1000 length=1000 color=0,1,0,0.2
place Vis z=0
fieldlines center=0,0,0 radius=500 square=1 exit=1 nLines=500 N=50
```

5.7 multipole

These plots of field lines show multipoles from dipole thru dodecapole, looking from the z-axis. The field lines look reasonable; to really appreciate this you must run it and manipulate the image in real time to look at it from different directions. The field lines are in 3-D, and these views do not accurately reflect their density.

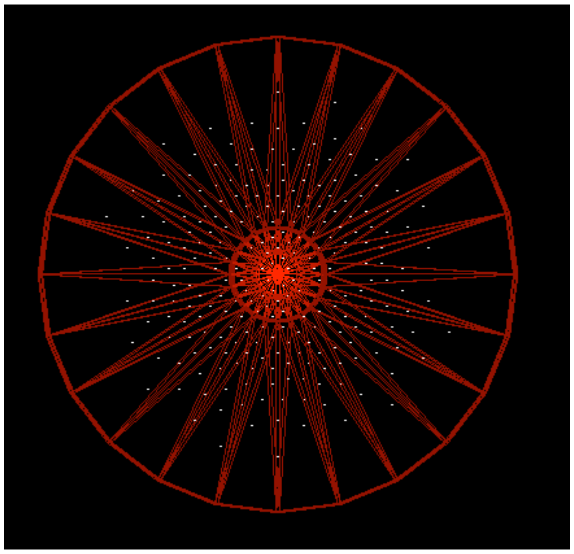
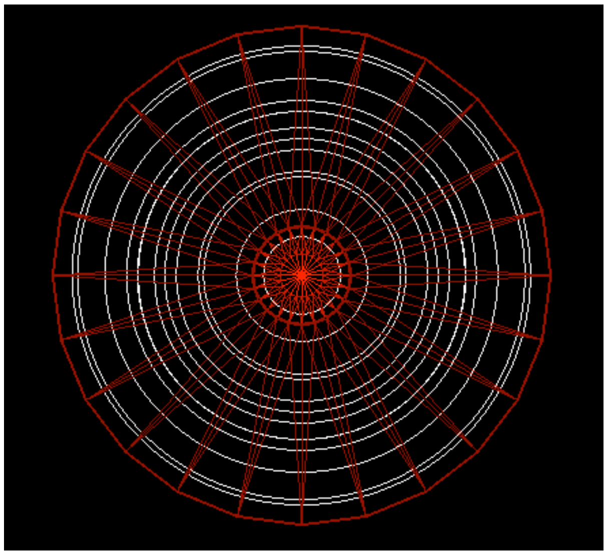







The input file for this test is multipole.g4bl:

```
*         multipol.g4bl multipole field lines
physics QGSP_BERT
#multipole MP fieldLength=200 ironLength=200 apertureRadius=200
ironRadius=400 \
    ironColor=0,0,1,.3 dipole=1
#fieldlines center=0,0,0 radius=200 square=1 dl=1 nLines=10 exit=1
#multipole MP fieldLength=200 ironLength=200 apertureRadius=200 \
   ironRadius=400 ironColor=0,0,1,.3 quadrupole=1
#param N=4 r1=80 r2=110 r3=160
#multipole MP fieldLength=200 ironLength=200 apertureRadius=200 \
   ironRadius=400 ironColor=0,0,1,.3 sextupole=1
#param N=6 r1=80 r2=110 r3=160
#multipole MP fieldLength=200 ironLength=200 apertureRadius=200 \
   ironRadius=400 ironColor=0,0,1,.3 octopole=1
#param N=8 r1=100 r2=120 r3=160
#multipole MP fieldLength=200 ironLength=200 apertureRadius=200 \
   ironRadius=400 ironColor=0,0,1,.3 decapole=10
#param N=10 r1=100 r2=120 r3=160
multipole MP fieldLength=200 ironLength=200 apertureRadius=200 \
   ironRadius=400 ironColor=0,0,1,.3 dodecapole=100
param N=12 r1=100 r2=120 r3=160
# generate 3 field lines, spaced 1/N around the circle
do i 0 $N-1
   if $i==$N-1
         param exit=1
   else
         param exit=0
   enddo
   param a=$i*2*3.14159/$N
   fieldlines nLines=0 dl=1 exit=$exit $r1*cos($a),$r1*sin($a),0 \
          $r2*cos($a),$r2*sin($a),0 $r3*cos($a),$r3*sin($a),0
enddo
place MP z=0
```

5.8 pillbox

These plots of E-field (left) and B-field (right) lines show a pillbox looking from the z-axis, at 90 degrees in the RF cycle (max fields). The field lines look reasonable; to really appreciate this you must run it and manipulate the image in real time to look at it from different directions). The inner red circle is a window for muon accelerators. The field lines are in 3-D, and these views do not accurately reflect their density.

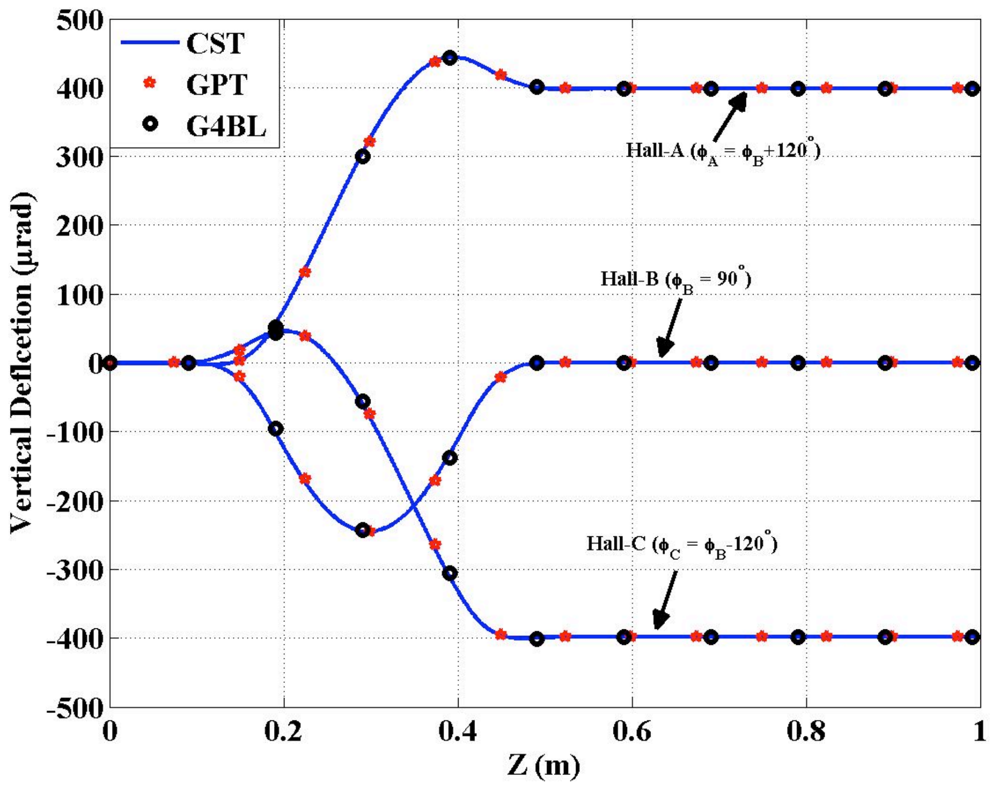
The input file for this test is pillbox.g4bl:

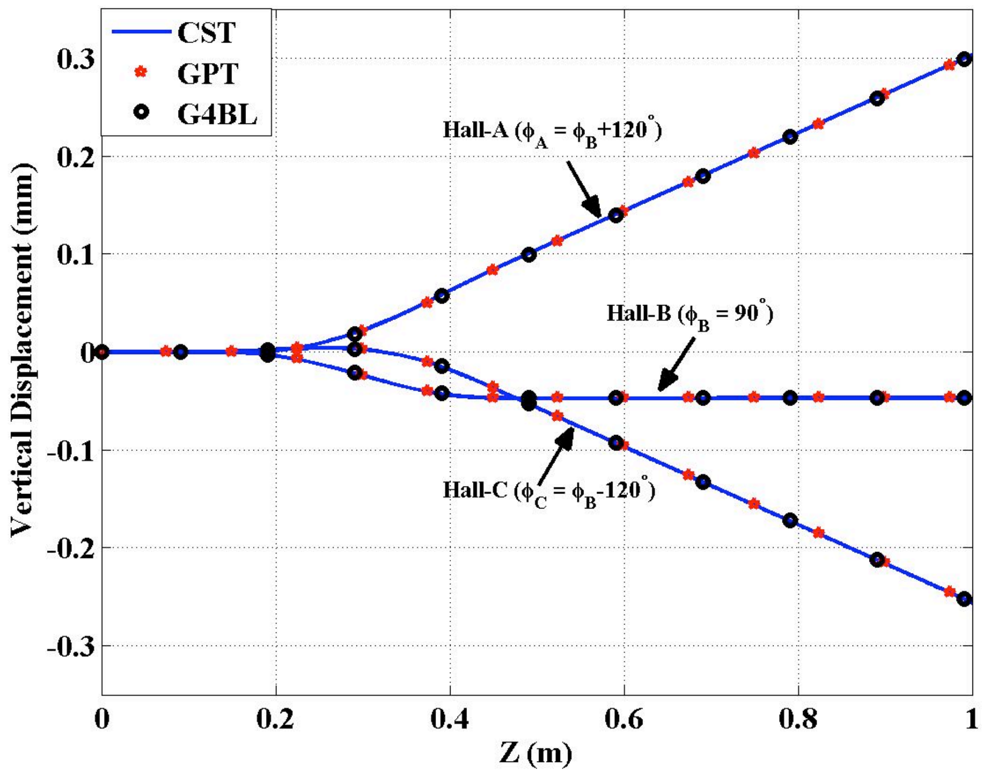
```
#        pillbox.g4bl
physics QGSP_BERT
pillbox Pbox maxGradient=10.0 frequency=0.20125 innerLength=430 phaseAcc=90
place Pbox z=0
fieldlines center=0,0,0 Efield=1 radius=800 exit=1 nLines=500
#fieldlines center=400,0,0 radius=800 exit=1 nLines=25
```

Test 2
The magnitude of the RF B-field has been compared to a pillbox modeled in Superfish by Milorad Popovic. The value was correct to 0.1%. This was comparing the maximum B-field for a given accelerating voltage in a given pillbox geometry. The details have been lost.

Test 3
Shahid Ahmed of JLab has tested deflecting/crabbing cavity beam dynamics studies and made a comparison of simulations with different codes. The figures show comparison of deflection and displacement of an 11 GeV electron passing through the axis of the superconducting deflecting cavity.

GPT : General particle tracer code
CST : CST microwave studio particle tracker

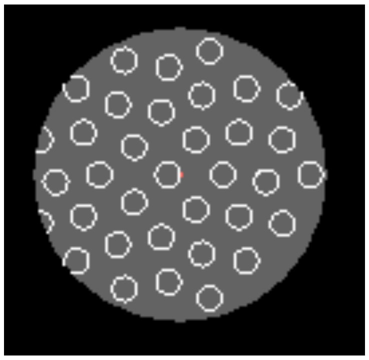

CST
GPT
G4BL
Hall-A ($\phi_A = \phi_B+120^\circ$)
Hall-B ($\phi_B = 90^\circ$)
Hall-C ($\phi_C = \phi_B-120^\circ$)
Vertical Deflcetion (μrad)
Z (m)
500
400
300
200
100
0
-100
-200
-300
-400
-500
0
0.2
0.4
0.6
0.8
1

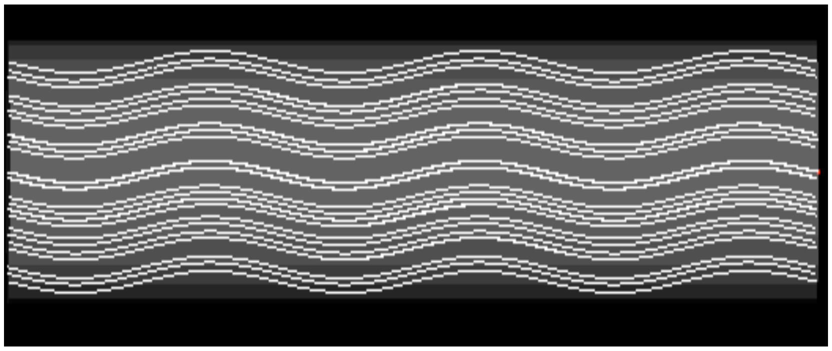

CST
GPT
G4BL
Hall-A ($\phi_A = \phi_B+120^\circ$)
Hall-B ($\phi_B = 90^\circ$)
Hall-C ($\phi_C = \phi_B-120^\circ$)
Vertical Displacement (mm)
Z (m)
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
0
0.2
0.4
0.6
0.8
1

5.9 helicaldipole

Test 1 – Field Lines

Note that the solenoid field is much larger than the helical dipole field, so the field lines do not follow the reference particle's much broader helical trajectory.





The input file for this test is helicaldipole.g4bl:

```
*	helicaldipole.g4bl
physics QGSP_BERT
helicaldipole HelicalDipoleField radius=320 length=3000.0 \
    model=1 Bsolenoid=5 bQ=0.0 bD=1.0 phi0=0.0 lambda=1000.0
place HelicalDipoleField z=1500
tubs Vis outerRadius=320 length=3000 color=1,1,1,.3
place Vis z=1500
fieldlines exit=1 center=0,0,1500 radius=320 nLines=20 dl=1
```

Test 2 – Comparison to Helical Solenoid

The helicaldipole command is intended to permit simulation of helical structures involving helical dipole, quadrupole, and sextupole fields. It simply generates these fields according to its parameters. A more realistic way to generate these fields is to use a large number of thin coils with their centers placed along an appropriate helix, which has been named "helical solenoid". Katsuya Yonehara compared these two systems, using the following parameters:

Parameter	Unit	Helical Solenoid magnet	*helicaldipole* module
κ		1.0	1.0
λ	mm	1000	1000

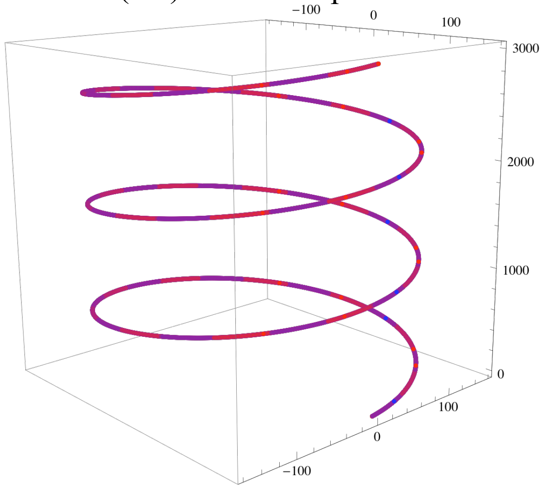
Parameters		HS coil length = 25 mm
		# of coils per λ = 20
		HS coil inner radius = 350 mm
		HS coil outer radius = 400 mm
		HS coil current = -256 A/mm^2
		Radius of coil center = 225 mm
			bd = 1.41412
			bQ = -0.210448
			bsol = -5.61552

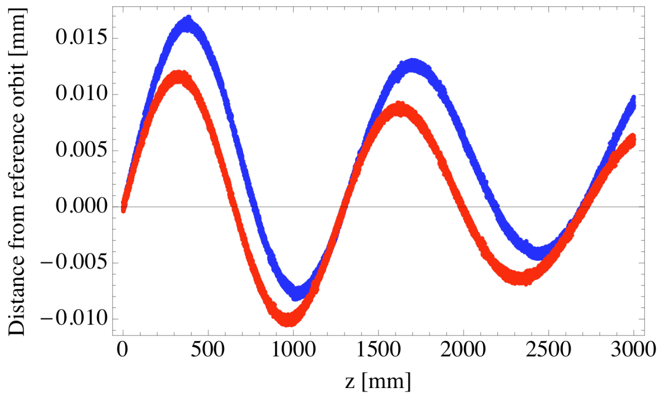
The following values of the two systems compare quite well:

Value	Unit	Helical Solenoid Magnet	*helicaldipole* module
b	T	-1.3084	-1.3093

b' (=∂b/∂ρ)	T/m	0.5349	0.5435
b/b'	m	-2.4460	-2.4089
bz	T	-4.3069	-4.3040
p (from analytical formula)	GeV/c	0.20233	0.20208
p (from particle tracking)	GeV/c	0.20234	0.20234
$\hat{D}$		1.4505	1.4436
η		0.4803	0.4775
Q_+		0.8747	0.8747
Q_-		0.7847	0.7856

The following plots compare the equilibrium orbits for *helicaldipole* (blue) to the helical solenoid (red). The discrepancies between the two systems are at the 0.01 mm level.





5.10 absorber

The absorber command uses polycones and tubs to implement a complex absorber geometry. Visualization and tracking are consistent with its definition. The details have been lost.

5.11 material

G4beamline materials use the NIST database implemented by the Geant4 collaboration. Here is a comparison of selected materials between G4beamline and the PDG "Particle Physics Booklet", July 2008:

Material	Density (g/cm^3)		Radiation Length (m)		Nucl. Interact. Length (m)	
	G4beamline	PDG	G4beamline	PDG	G4beamline	PDG
LH2	0.07080	0.071	8.904	8.879	4.982	7.324
LiH	0.820	0.820	0.971	0.971	0.732	0.830
He	0.000166	0.000166	5,671	5,682	3,343	4,277
Li	0.534	0.534	1.550	1.550	1.252	1.335
Be	1.848	1.848	0.353	0.353	0.395	0.421
C	2.000	2.210	0.213	0.193	0.401	0.193
Al	2.699	2.699	0.0890	0.0889	0.389	0.397

Fe	7.874	7.874	0.0176	0.0176	0.170	0.167
Pb	11.350	11.350	.00561	.00561	0.182	0.175
U	18.950	18.950	.003166	.003166	0.114	0.110

The densities and radiation lengths are consistent, but the nuclear interactions lengths differ substantially. It's not clear what this means. Note that G4beamline (Geant4) tracking does not use the nuclear interaction length directly, instead it uses a more accurate and much more complicated technique based on cross-sections and densities.

The input file for this test is material.g4bl:

```
material Ignore density=1 LH2,0.1 He,0.1 Li,0.1 LITHIUM_HYDRIDE,0.1 C,0.1 \
        Be,0.1 Al,0.1 Fe,0.1 U,0.1 Pb,0.1
material
```

5.12 cosmicraybeam

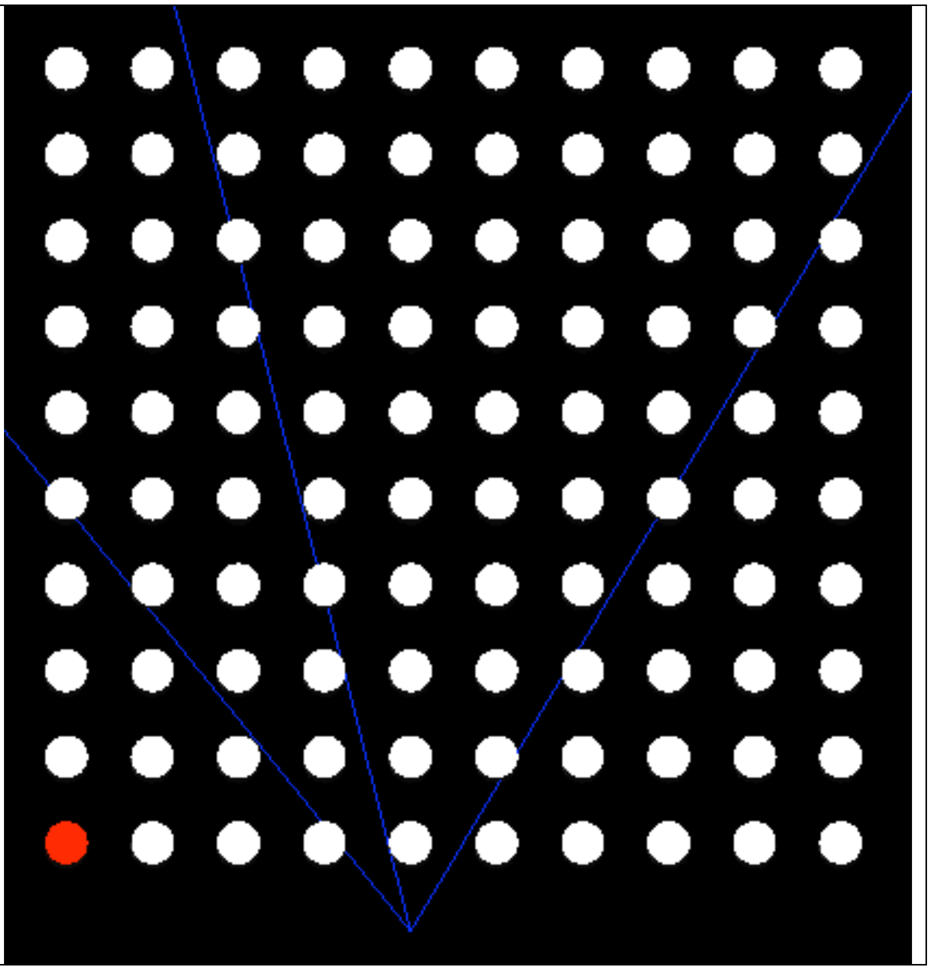
This command does not reproduce recent measurements very well (factors of 2-4), and needs to be completely re-done.

6 Visualization

Visualization has been used extensively during the development and application of G4beamline. Literally thousands of images have been examined. Most users use the Open Inventor viewer, but the other supported viewers have all been tested and used at least occasionally. The wireframe mode in Open Inventor sometimes renders an object in reduced wireframe mode (i.e. only real lines are displayed, no surface-tracing triangles are used). In addition, Boolean operations between solids sometimes display incorrectly (a well-known bug in the Geant4 visualization system, which is being fixed). In no cases have the locations or sizes of objects been rendered incorrectly.

Test 1
As a test of visualization consistency with tracking, a 10-by-10 array of cylinders was constructed, and tracks were sent into it. The cylinders hit by the tracks are identified by the steppingVerbose output, which can be compared to the picture below (1,1 is lower left in red, 10,10 is upper right). Careful examination in the viewer showed which cylinders were hit and which were missed, in agreement with the steppingVerbose output from tracking.

Track 1	Track 2	Track 3
4,1	6,2	5,1
2,4	7,3	4,3
1,5	8,5	4,4
	9,7	3,7
	10,8	3,8
Missed	**Missed**	**Missed**
2,3	6,1	4,5
	7,4	3,9
	9,6	



The input file for this test is visualization.g4bl:

```
#	visualization.g4bl
physics QGSP_BERT
beam gaussian sigmaXp=0.5 nEvents=1000
param color=1,0,0 steppingVerbose=1
cylinder Cyl outerRadius=25 length=200
do i 1 10
	param x=$i*100-500
	do j 1 10
		param z=$j*100
		place Cyl z=$z x=$x rotation=X90 rename=$i,$j color=$color
		param color=1,1,1
	enddo
enddo
```

7 References

[1] http://g4beamline.muonsinc.com

[2] http://geant4.cern.ch

[3] http://root.cern.ch

[4] http://wwwasd.web.cern.ch/wwwasd/lhc++/clhep/index.html

[5] F. James, Comp. Phys. Comm. 60 (1990) 329.

[6] http://arxiv.org/abs/hep-ex/0512005

[7] http://pdg.lbl.gov/2010/reviews/rpp2010-rev-passage-particles-matter.pdf

[8] http://mu2e.fnal.gov/

[9] T. J. Roberts, "Analysis of Geant4 Physics Processes for μ^- Capture at Rest", http://mu2e-docdb.fnal.gov:8080/cgi-bin/ShowDocument?docid=1119

[10] http://www-ap.fnal.gov/MARS/

[11] http://www.fftw.org/

[12] Qiang et al, Phys. Rev. STAB, **9**, 044204 (2006).
Hockney and Eastwood, Computer Simulation Using Particles, Hilger, 1988, section 6-5-4, p200.

[13] CERN Courier 50, no. 1, p10 (2010).