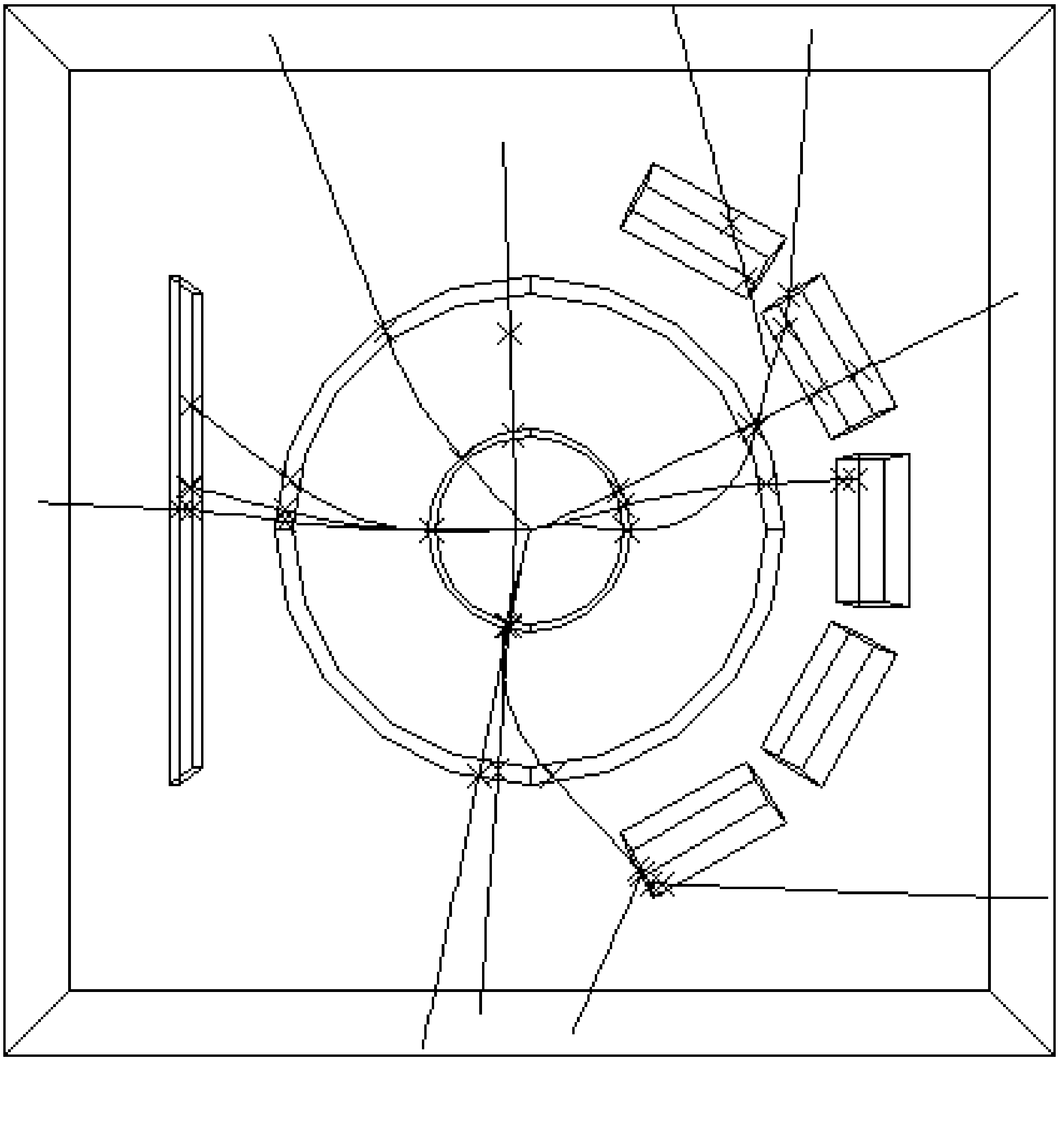
Particle Physics Simulators

- Used extensively by experimental nuclear and high-energy physicists in both detector design and study
- Simulate particle trajectories through detector apparatus, incorporating relevant physics processes
- Can determine complicated detector performance (e.g. efficiency, acceptance) where not analytically possible
- Great tool for learning particle physics!

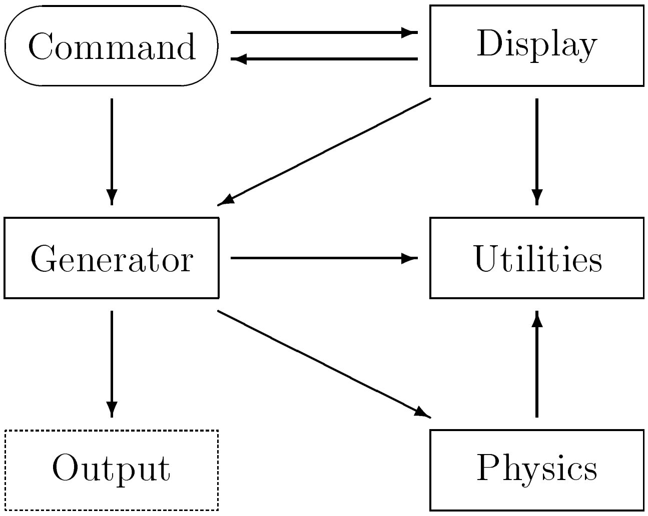


“ABC” Physics Simulator - Goals

- Approximate physical processes common to most particles in transport through media:
  - curvature in magnetic fields
  - particle decay
  - multiple scattering
  - energy loss
- Modularize code to accommodate additional user-defined processes, if desired
- Read input from command-line or script files, avoiding need to recompile
- Generate graphical display to visualize apparatus and particle trajectories
- Output specific “detector hit” data for quantitative analysis (e.g. with plotting package)
- Keep code compact and simple enough so a user can retrieve, compile, and be running the simulator in 5 minutes!
  - available on “wwwrel.ph.utexas.edu” in:

    /Public/Students/ambrose/abc
  - consists of single header file, nine “C” routines, several example scripts, and the Makefile

"ABC" - Outline of code



- The $\boxed{\text{Command}}$ subroutine reads and lists data (e.g. objects, particles, and other definitions) and sets process flags.
- $\boxed{\text{Display}}$ opens the graphics window and draws the apparatus, allowing the user to adjust view angles, scales, and toggle settings (axes, color, perspective).
- The $\boxed{\text{Generator}}$ routine produces particles from a pre-defined list of intitial conditions, and "swims" the tracks through the detector. Particles intersecting detector objects produce hits which are sent to standard $\boxed{\text{Output}}$.
- $\boxed{\text{Physics}}$ routines apply physical processes (e.g. energy loss, decay) to tracks, updating the current step or producing new particles.
- The $\boxed{\text{Utilities}}$ routine contains various functions used by the other routines, such as vector rotations, line-object intersection, and random number generation.

"ABC" - Data Input

- Define colors for the $\boxed{\text{Display}}$ routine (0,1):

  `c` <num> <red> <green> <blue>

- Define particle data (with optional decay modes):

  `p` <num> <name> <color> <charge> <mass> <$c\tau_0$> <decay>
  ...<mode> <branch>

- Define materials used for objects:

  `m` <num> <name> <A> <Z> <ρ> <X_0>

- Define universe (object 0):

  `u` <color> <mat> <x(min,max)> <y(min,max)> <z(min,max)>

- Object data:

  `o` <num> <name> <shape> <col> <mat> <det> <B_{typ}> <B_{mag}>
  ...<pos(x,y,z)> <siz(1,2,3)> <rot(α, β, γ)>

- Initial conditions for the $\boxed{\text{Generator}}$ routine:

  `g` <num> <prt> <flag> <pos(x,y,z)> <mom(x,y,z)> <Δp>

"ABC" - Script Files

- Default file *abc.s* is read upon startup, which contains typical particles and materials:

```
# Define colors: (color 0 pre-defined as black or white)
#   num  red   grn   blu
  c 01   1.0   0.0   0.0
  c 02   0.0   1.0   0.0

  c 98   0.0   0.0   0.0 # black
  c 99   1.0   1.0   1.0 # white

# Define particles:
#   num/name     col  chrg  mass(GeV)  life(m)  decay
  p 01 photon    01   +0.0  0.0        0.0      0
  p 02 neutrino  02   +0.0  0.0        0.0      0

  p 18 sigma+    03   +1.0  1.118937   0.0240   2
       001009 0.5164                            # p pi0
       001108 0.4836                            # n pi+
  p 19 sigma0    04   +0.0  1.119255   2.2e-11  1
       001601 1.0                               # lambda gamma

# Define materials: (material 0 pre-defined as vacuum)
#   num/name      A        Z      den(g/cm2)   rad(cm)
  m 01 air        14.610   7.30   1.205e-3     30423
  m 02 helium     4.0000   2.00   0.125        755.0

  m 06 lead       207.19   82.0   11.35        0.560
  m 07 uranium    238.03   92.0   18.95        0.320

# Input experiment scripts:
  i example
```

- Script files may call other script files (up to 3 deep). The file *example.s* provides a template for creating new apparatus.

Display Routine

- The drawing routine performs an Euler rotation and translation of object frame coordinates to the lab frame, and then a second rotation to the view frame, applying optional perspective using view frame z-position and view distance (d_v):

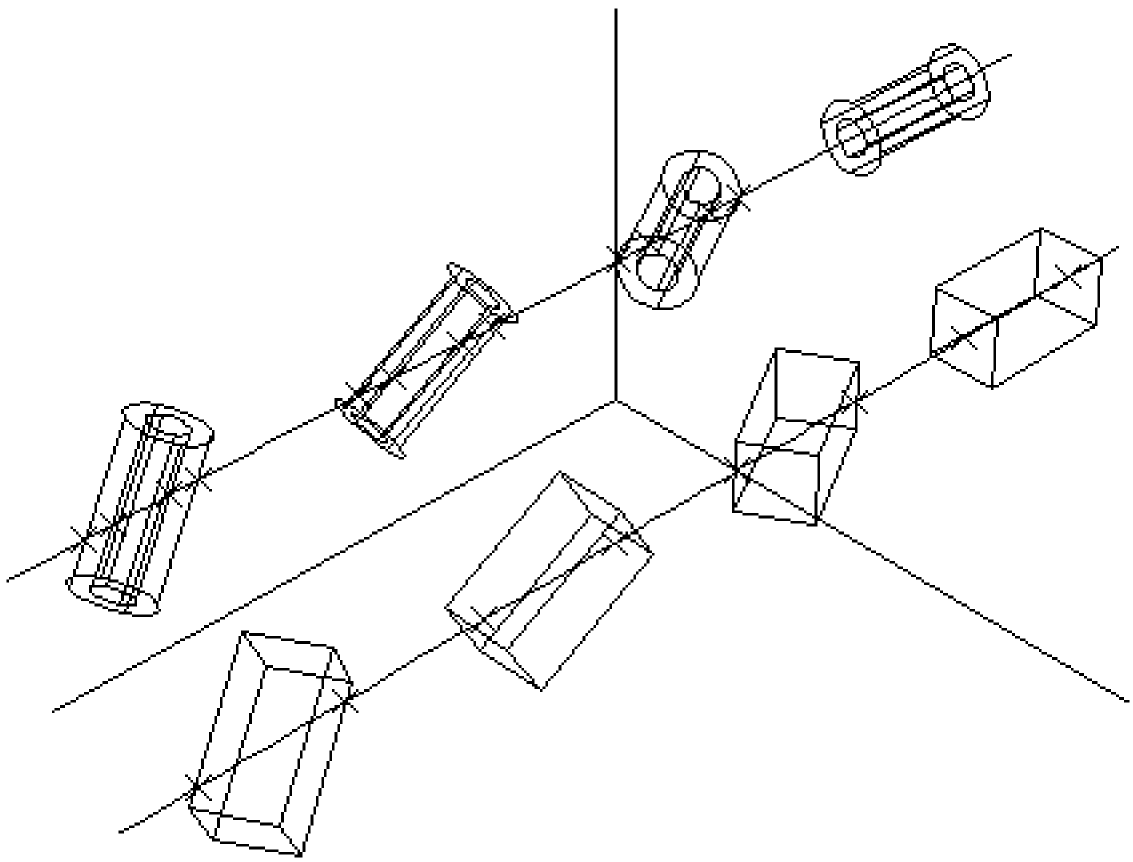
$$x_v{}' = x_v \frac{d_v}{d_v - z_v} \ , \ y_v{}' = y_v \frac{d_v}{d_v - z_v}$$

- Two objects (box,cylinder) are defined. Each object needs three function calls:

  i) drawing function

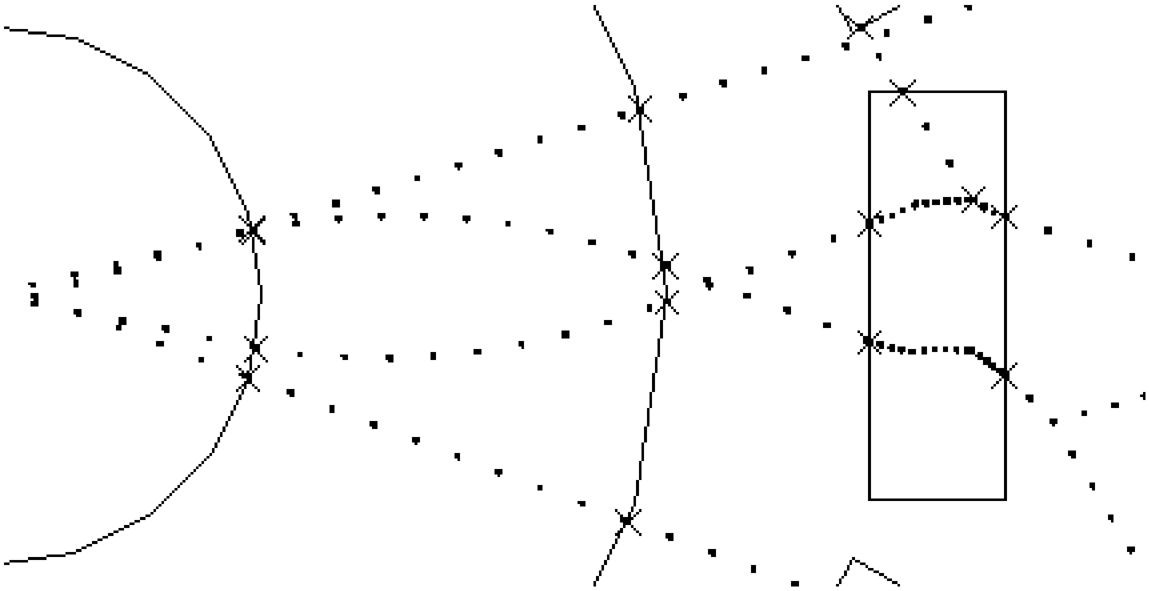
  ii) point location (is point inside or outside?)

  iii) line intersection on object



Generator Routine

- Initialize first step with input position/momentum. If Δp selected, vary momentum magnitude with normal deviate. If generator flag set, force decay.
- Loop over tracks from current to total number. Swim tracks until momentum falls below threshold, or decay occurs:
  - Fill initial step data, determine object location
  - Choose step size from object field strength and radiation length X_0
  - Take trial step, checking for intersection with new object or exit of current one, and truncate
  - Apply physics processes to final step point:
    * curvature in magnetic field
    * multiple scattering
    * energy loss
    * particle decay (add new tracks to queue)
  - Output position/momentum to stdout if entering or leaving detector object and fill hit array for $\boxed{\text{Display}}$



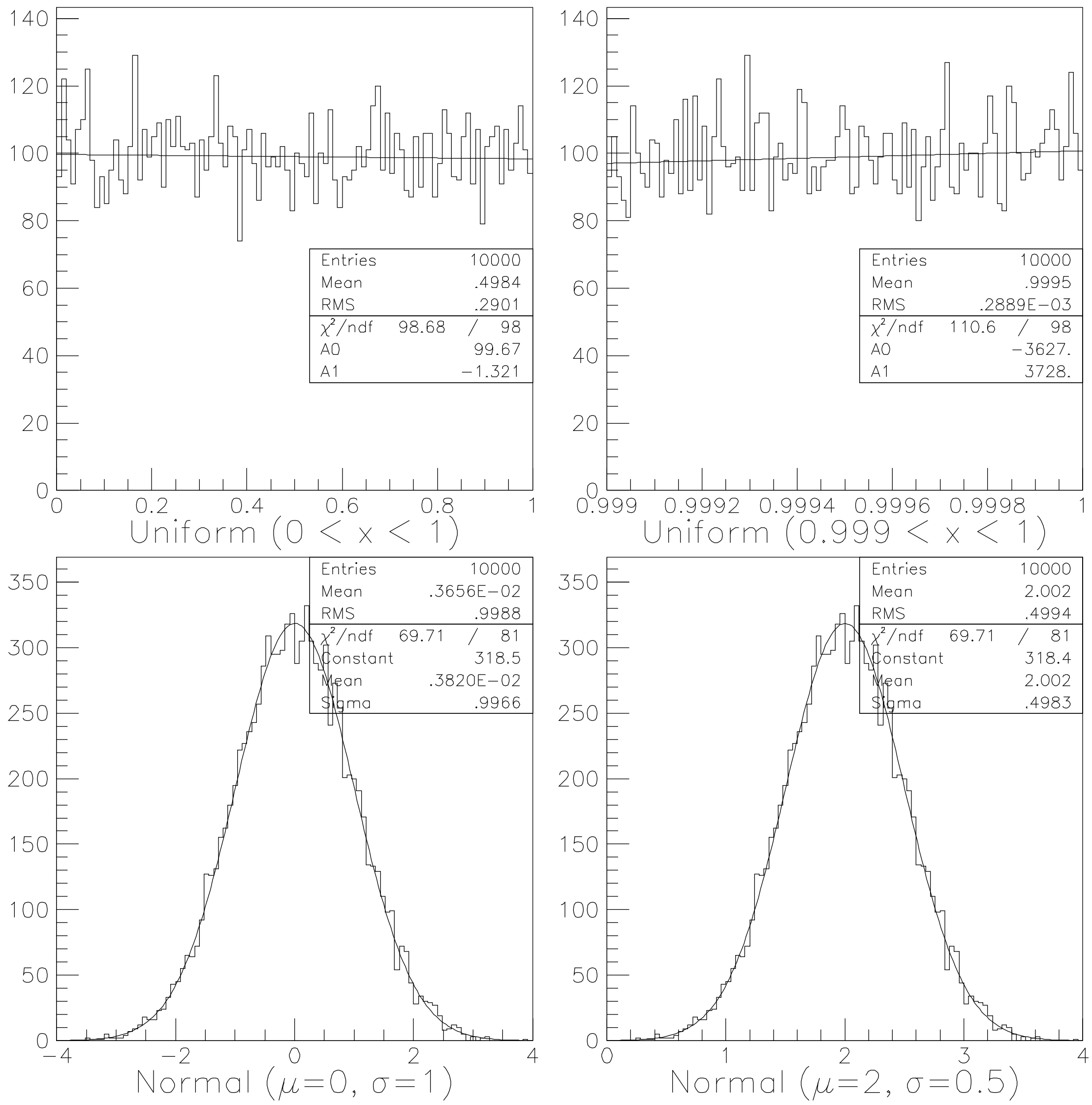
Random Variable Generation

- Uniform variables chosen with $< math.h >$ function:

```
double drand48()
```

- Normal variables chosen using two uniform deviates (r_1, r_2) and the Box-Muller method [Numerical Recipes (7.2.10)]:

$$
\begin{aligned}
r_n &= \sin(2\pi r_1)\sqrt{-2\ln r_2} \\
r_n &= \mu + \sigma r_n
\end{aligned}
$$

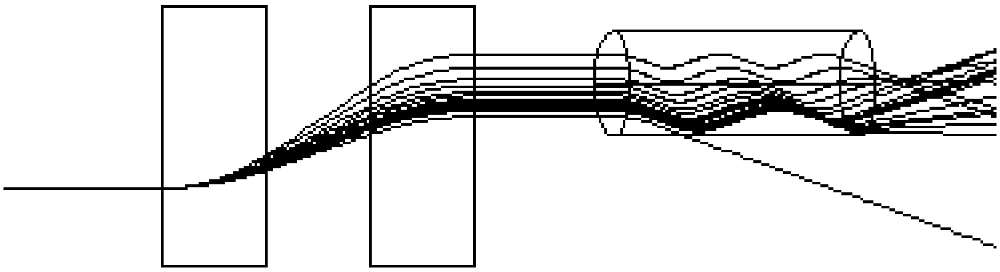


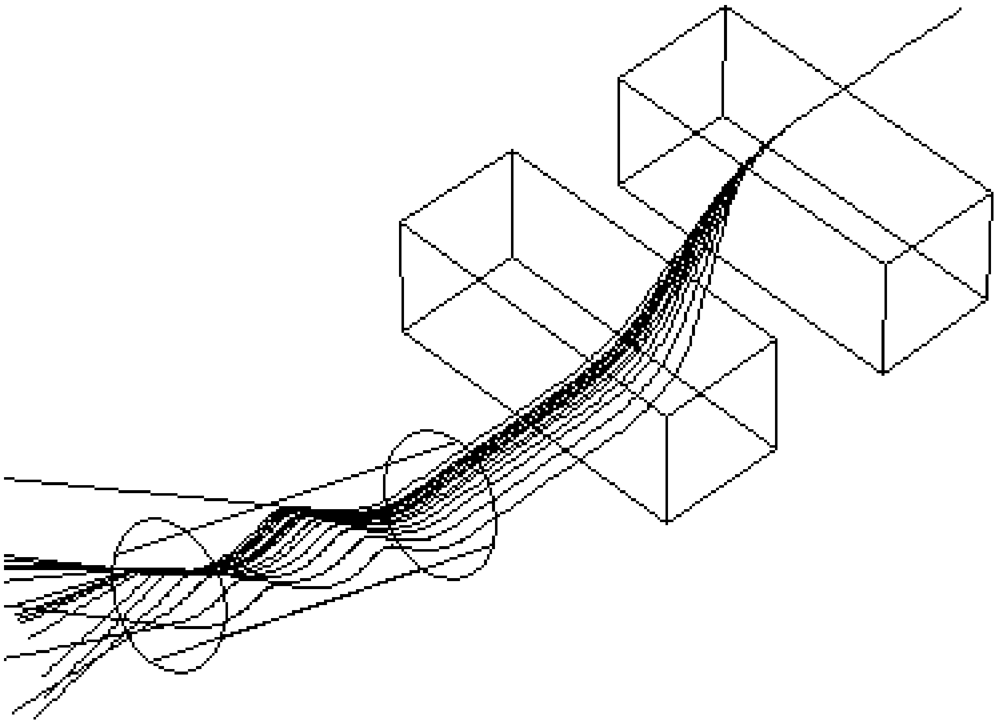
Applying Magnetic Fields

- In uniform, static B-fields, charged particles follow helical paths with a radius of curvature:

$$a \sim \frac{c\,p_\perp}{z\,B} \;\Rightarrow\; p_\perp \equiv \textit{ perpendicular momentum}$$

- Defining "dipole" objects with uniform B along z-axis, step coordinates and momenta are rotated to object frame, separating $p_\perp$ and $p_\parallel$. Final step perpendicular position and momentum are adjusted to follow curvature, then rotated back to lab frame.





Applying Particle Decay

- For particle with mass and energy (m, E), the dilated lifetime is $(\tau = \gamma\tau_0, \gamma = E/m)$, and survival probability over $(\Delta t = t_2 - t_1)$ obeys Poisson statistics with $(y = 0, \lambda = \Delta t/\tau)$:

  $$P(y) = \frac{\lambda^y}{y!}e^{-\lambda} = e^{-\Delta t/\tau}$$

  A uniform deviate (r_p) is chosen and, if $(r_p > P(y))$, a decay is generated.

- The partial decay rate for mass m (in rest frame) into n bodies is a function of the matrix element $\mathcal{M}$ and phase-space $d\Phi_n$:

  $$d\Gamma = \frac{2\pi^4}{2m} \mid \mathcal{M} \mid^2 d\Phi_n(P; p_1, p_2, \ldots, p_n)$$

  Approximate $d\Gamma$ by ignoring $\mathcal{M}$ and choosing $d\Phi_n$ over uniform phase-space, then boosting back to lab frame.

- For two-body decay, final-state momenta are pre-determined from conservation of 4-momentum:

  $$\mid \mathbf{p_1} \mid = \mid \mathbf{p_2} \mid = \frac{[(m^2 - (m_1 + m_2)^2)(m^2 - (m_1 - m_2)^2)]^{1/2}}{2m}$$

  The phase-space element then reduces to:

  $$d\Gamma \sim \frac{\mid \mathbf{p_1} \mid}{m^2} d\Omega \;\Rightarrow\; d\Omega = d\phi_1 d(\cos\theta_1)$$

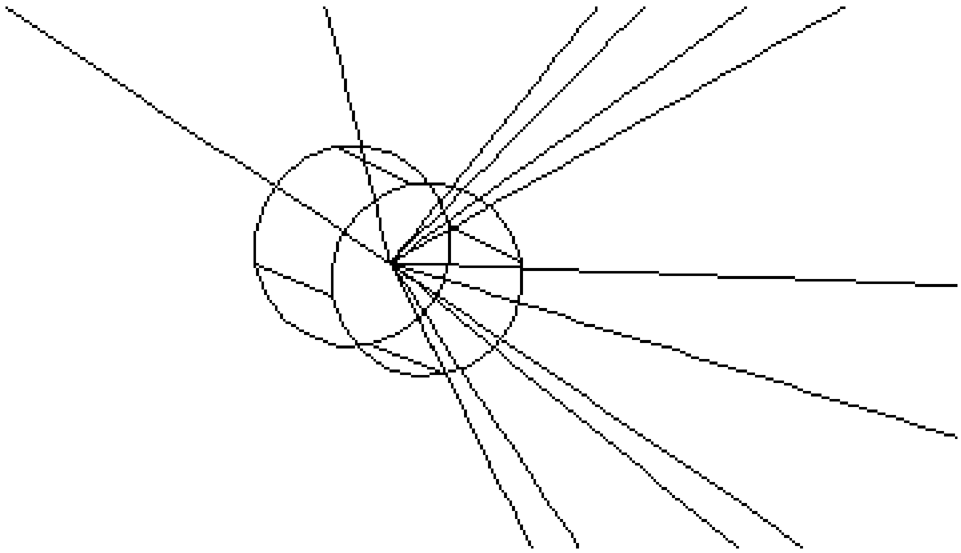
  where a uniform $\phi_1(-\pi, \pi)$ and $\cos\theta_1(-1, 1)$ are chosen.

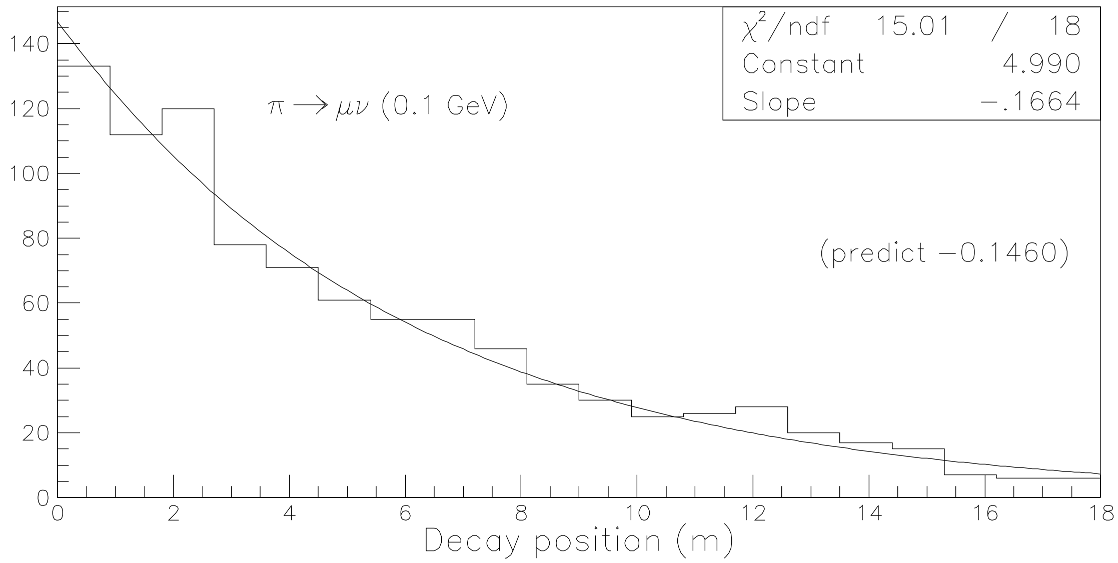
- Three-body decays form a unique production plane, where a Dalitz plot of any two final-state energies will be uniform. Choosing two such energies E_i $(m_i < E_i < (E - m_j - m_k))$, the phase-space becomes:

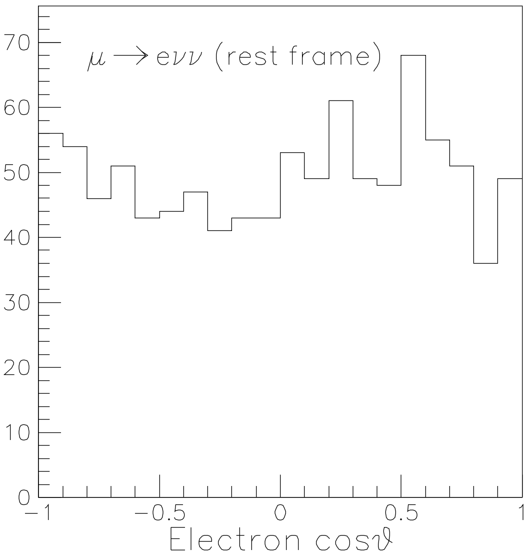
  $$d\Gamma \sim dE_1\, dE_2\, d\Omega \;\Rightarrow\; d\Omega = d\alpha\, d(\cos\beta)\, d\gamma$$

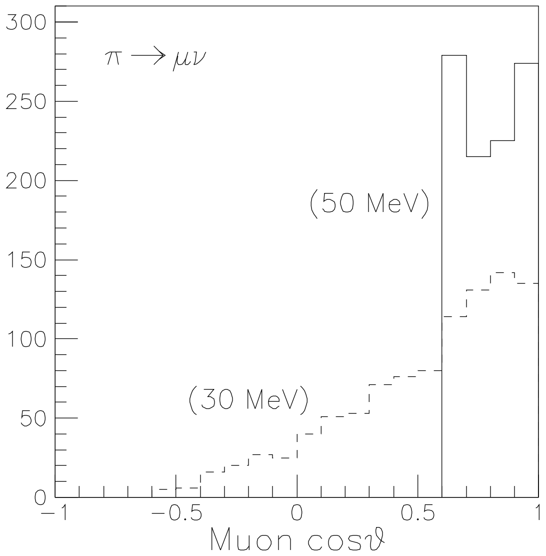
  where a uniform $\alpha(-\pi, \pi)$, $\cos\beta(-1, 1)$, and $\gamma(-\pi, \pi)$ are chosen.

Testing Particle Decay









Applying Particle Energy Loss

- Charged particles heavier than electrons lose energy within a material primarily through inelastic collisions with atomic electrons, given by the Bethe-Bloch equation:

$$-\frac{dE}{dx} \;=\; K\,z^2\,\frac{Z}{A}\,\frac{1}{\beta^2}\;\left[\frac{1}{2}\ln\frac{2m_ec^2\beta^2\gamma^2T_{max}}{I^2} - \beta^2 - \frac{\delta}{2} - \frac{C}{Z}\right]$$

  where the maximum kinetic energy transfer to electrons is:

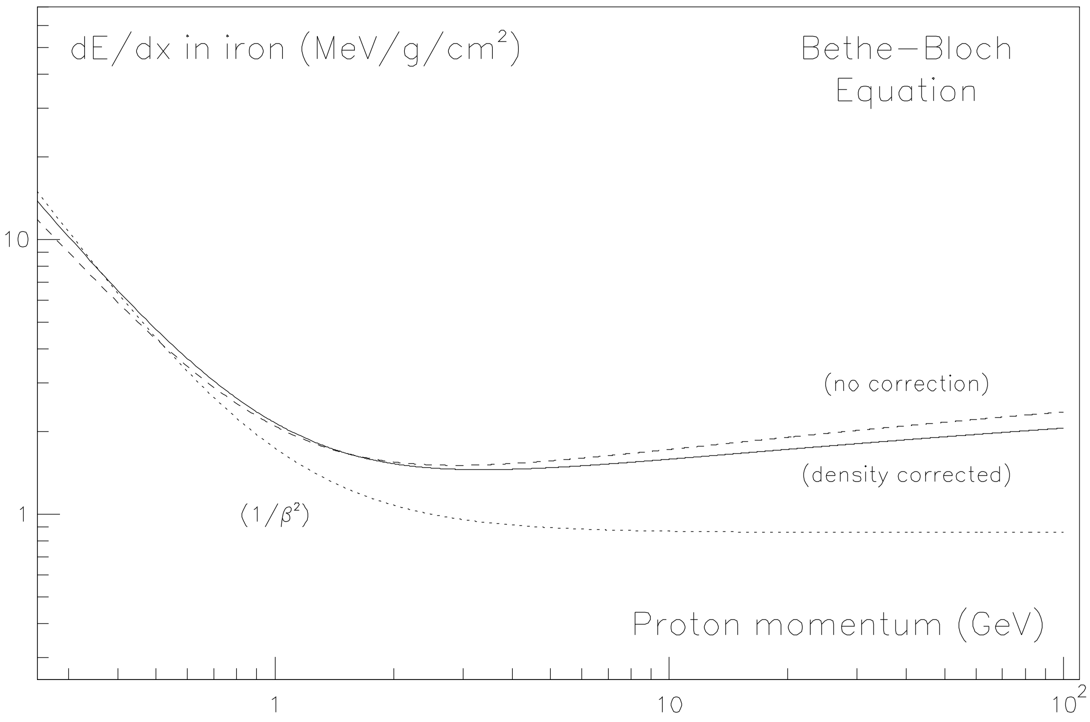
$$T_{max} \;=\; \frac{2m_ec^2\beta^2\gamma^2}{1+2\gamma(m_e/m)+(m_e/m)^2}$$

- For a material with atomic number and mass (Z, A), the mean excitation potential $I\ (in\,eV)$ can be approximated with:

$$I = \begin{cases} 12\,Z + 7 & (Z < 13) \\ 9.76\,Z + 58.8\,Z^{-0.19} & (Z \geq 13) \end{cases}$$

- Ignoring the low-energy shell correction C, the density correction δ can be approximated (for high-energy) with:

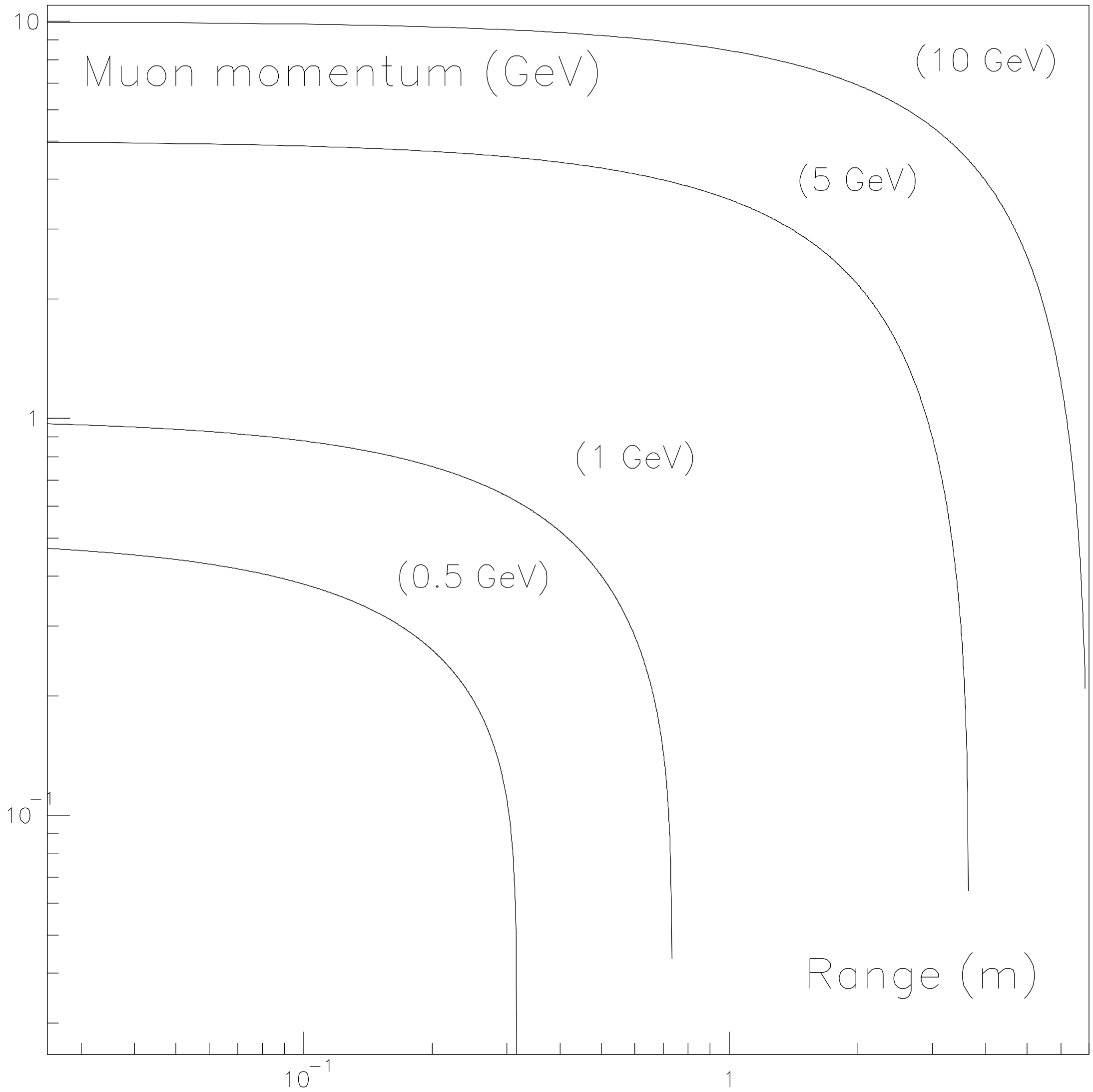
$$\frac{\delta}{2} \;\sim\; \ln\left(\hbar\omega_p/I\right) + \ln\beta\gamma - 1/2 \;\Rightarrow\; \hbar\omega_p = 28.816\sqrt{\rho\,Z/A}$$



Testing Energy Loss and Range

- The particle energy loss is calculated for each step by integrating dE/dx over the step length using a fourth-order Runge-Kutta scheme, until the momentum falls below a pre-determined threshold ($\sim 10\,MeV$).
- Testing the integration with muons in iron, the calculated range averages about 5% of tabulated values.

Momentum(GeV)	*Range(m)*	*Expect(m)*	*Error(%)*
0.1	0.0128	0.0121	-5.8
0.5	0.3165	0.3370	+6.0
1	0.7320	0.7400	+1.0
5	3.6400	3.3800	-7.7
10	6.9200	6.7300	-2.8

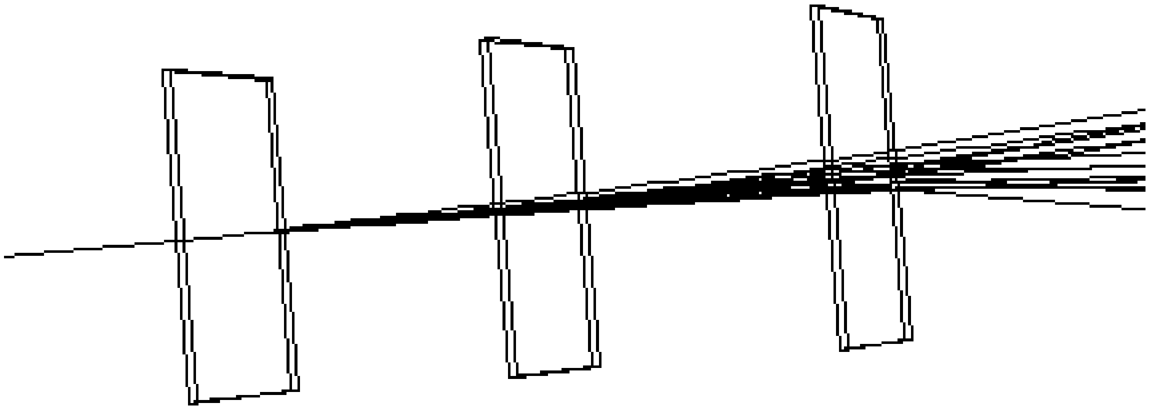


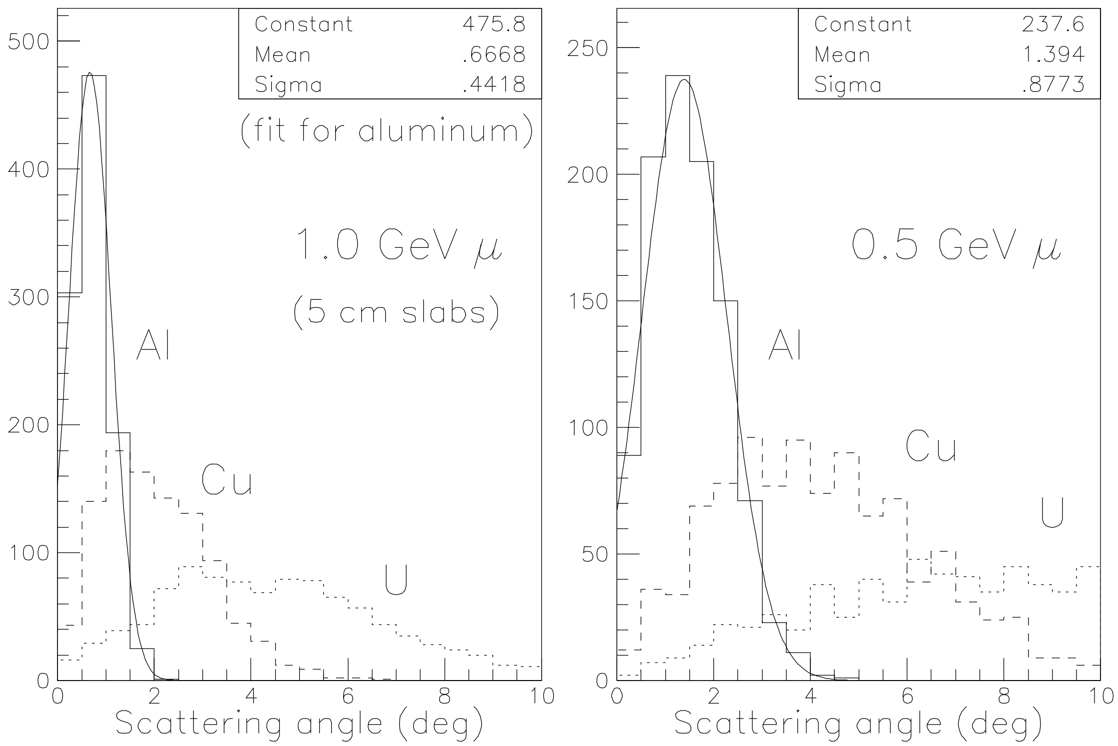
Applying Multiple Coulomb Scattering

- Charged particles traversing a material suffer many small-angle scatters from collisions with nuclei. The rms plane scattering angle (through pathlength x) is approximated by:

$$\theta_0 = \frac{13.6 MeV}{\beta\, c\, p}\, z\, \sqrt{x/X_0}\; [1 + 0.038\, \ln x/X_0]$$

- Assuming a normal distribution of θ_{plane}, two normal rv's are chosen with ($\sigma = \theta_0$), so the random space scatter ($\theta_s = \sqrt{\theta_1^2 + \theta_2^2}$) with a uniform random azimuthal angle ϕ_s.





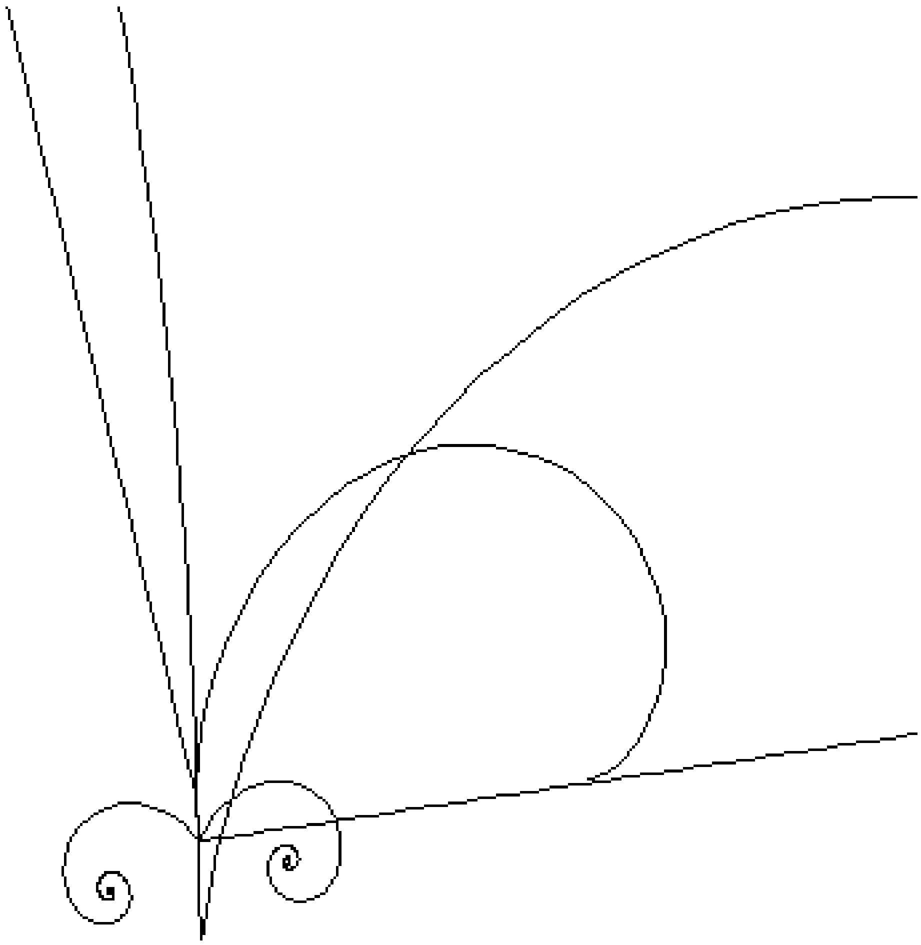
“ABC” Physics Simulator - Examples

- Reproduce famous physics experiments!

Example - Discovery of the Ω^-

- Mimic the BNL 80-inch bubble chamber with a box containing liquid hydrogen and $0.4T$ dipole field.
- Force Ω^- decays (assume creation by $K^- + p \to \Omega^- + K^+ + K^0$), for example:

$$\begin{array}{lllll} \Omega^- \to \Xi^0\,\pi^- \;, & \Xi^0 & \to & \Lambda^0\,\pi^0 \\ & \pi^0 & \to & e^+\,e^-\,\gamma \\ & \Lambda^0 & \to & p\,\pi^- \\ & \pi^- & \to & \mu^-\,\nu \end{array}$$



Example - Detection of η meson